BAB IV
IMPLEMENTASI DAN PENGUJIAN SISTEM

Pada bab ini akan dilakukan implementasi dan pengujian terhadap *game* yang dibangun. Tahapan ini dilakukan setelah analisis dan perancangan selesai dilakukan dan selanjutnya akan diimplementasikan ke dalam bahasa pemrograman. Setelah implementasi maka dilakukan pengujian terhadap *game*. *Game* yang telah dibangun akan diimplementasikan untuk mengetahui apakah sistem di dalam *game* tersebut dapat berjalan sesuai dengan tujuannya atau tidak.

IV.1 Implementasi

Tujuan implementasi adalah untuk menerapkan perancangan yang telah dilakukan terhadap sistem sehingga pengguna dapat memberi masukan demi berkembangnya sistem yang telah dibangun sebagai simulasi dari *game action* 3D *The Walker*. Perangkat keras dan perangkat lunak yang digunakan untuk melakukan implementasi dan pengujian *game* adalah sebagai berikut:

a. Perangkat keras

   Perangkat keras yang digunakan untuk pembangunan *game* adalah sebagai berikut.

   1). Prosesor yang setara dengan kecepatan 2.0 GHz.
   2). RAM 2 GB.
   3). Harddisk dengan ruang kosong 10 GB.
   4). VGA Card shared 128 MB.
   5). Monitor dengan resolusi 1024 x 768 pixels.
   6). *Speaker.*
   7). *Keyboard* dan *mouse.*

b. Perangkat lunak

   Perangkat lunak yang digunakan untuk mengimplementasikan *game* adalah sebagai berikut.

1). Sistem Operasi Microsoft Windows 7 Ultimate 32bit

2). *Game engine Unity* 3D, dengan bahasa pemrograman berupa c# dan *javascript*.

3). *Rendering* gambar 3D menggunakan *3DStudio Max*, dan gambar 2D menggunakan *Adobe Photoshop CS3*, sedangkan untuk suara atau *sound effect* menggunakan *Adobe After Effects*.

4). *DirectX* 9

Sistem operasi harus sudah terpasang pada komputer, agar *game* yang dibangun dapat dijalankan pada komputer tersebut. Jika pengguna ingin membuka *project*nya maka terlebih dahulu harus memasangkan *Unity* 3D dan komponen-komponen yang mendukungnya. *File* TheWalker.exe dapat digunakan untuk mengimplementasikan *game* yang telah dibangun.

IV.1.1 Implementasi Antarmuka

Adapun implementasi antarmuka pada aplikasi *game The Walker* adalah sebagai berikut:

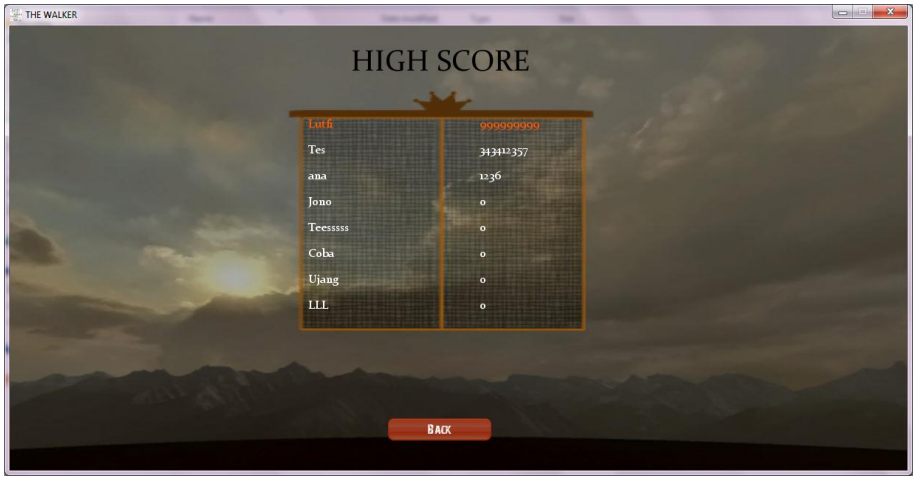
1. Antarmuka Menu Utama

Antarmuka menu utama merupakan tampilan menu utama pada saat pertama kali pengguna membuka aplikasi *game*. Implementasi antarmuka untuk perancangan *game start menu* yang dibangun dapat dilihat dari gambar IV.1.



Gambar IV. 1 Implementasi *Start Menu*

Di dalam *game* ini terdapat beberapa menu yang terbagi menjadi lima yaitu menu *start game*, *high score*, *how to play*, *credits*, dan *exit*. Menu *start game* berguna untuk memulai memainkan *game*, menu *high score* yaitu untuk menampilkan halaman *high score* dapat dilihat pada gambar IV.2, menu *how to play* yaitu untuk menampilkan halaman bantuan untuk pengguna memainkan *game* dapat dilihat pada gambar IV.3, menu *credits* yaitu untuk menampilkan halaman informasi pembangun *game* dapat dilihat pada gambar IV.4, dan terakhir menu *exit* digunakan untuk keluar dari aplikasi *game* dan memunculkan pesan ingin keluar game atau tidak dapat dilihat pada gambar IV.5.



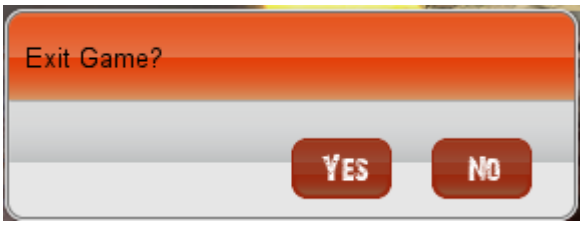
Gambar IV. 2 Implementasi *High Score*



Gambar IV. 3 Implementasi *How To Play*



Gambar IV. 4 Implementasi *Credits*



Gambar IV. 5 Implementasi Pesan

2. Antarmuka Mulai Permainan

Antarmuka mulai permainan merupakan tampilan ketika pengguna mengklik tombol *start game*. Pertama adalah tampilan dari antarmuka *scene video* awal main dapat dilihat pada gambar IV.6.



Gambar IV. 6 Implementasi *Scene Video* Awal Main

Tampilan antarmuka *scene video* untuk penjelasan *level* 1 dapat dilihat pada gambar IV.7



Gambar IV. 7 Implementasi *Scene Video Level* 1

Tampilan antarmuka memainkan *game* pada *level* 1 dapat dilihat pada gambar IV.8



Gambar IV. 8 Implementasi Memainkan *Level* 1

Tampilan antarmuka *mission* pada *level* 1 dapat dilihat pada gambar IV.9



Gambar IV. 9 Implementasi *Mission Level* 1

Tampilan antarmuka *scene video* untuk penjelasan *level* 2 dapat dilihat pada gambar IV.10



Gambar IV. 10 Implementasi *Scene Video Level* 2

Tampilan antarmuka memainkan *game* pada *level* 2 dapat dilihat pada gambar IV.11



Gambar IV. 11 Implementasi Memainkan *Level* 2

Tampilan antarmuka *mission* pada *level* 2 dapat dilihat pada gambar IV.12



Gambar IV. 12 Implementasi *Mission Level* 2

Tampilan antarmuka *scene video* untuk penjelasan *level* 3 dapat dilihat pada gambar IV.13



Gambar IV. 13 Implementasi *Scene Video Level* 3

Tampilan antarmuka memainkan *game* pada *level* 3 dapat dilihat pada gambar IV.14



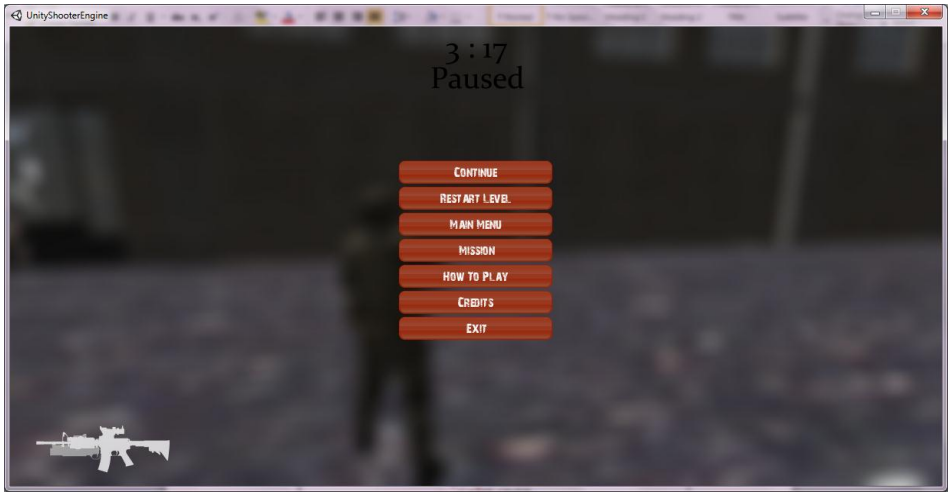
Gambar IV. 14 Implementasi Memainkan *Level* 3

Tampilan antarmuka *mission* pada *level* 3 dapat dilihat pada gambar IV.15



Gambar IV. 15 Implementasi *Mission Level* 3

Tampilan antarmuka memberhentikan *game* dapat dilihat pada gambar IV.16



Gambar IV. 16 Implementasi Memberhentikan *Game*

Tampilan antarmuka *mission complete* dapat dilihat pada gambar IV.17



Gambar IV. 17 Implementasi *Mission Complete*

Tampilan antarmuka *game over* dapat dilihat pada gambar IV.18



Gambar IV. 18 Implementasi *Game Over*

IV.2 Pengujian

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

IV.2.1 Pengujian *Alpha*

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

IV.2.1.1 Skenario Pengujian Aplikasi *Game*

Skenario pengujian menjelaskan pengujian terhadap sistem yang ada pada aplikasi *game The Walker*. Skenario pengujian dari *game The Walker* dapat dilihat pada tabel IV.1

Tabel IV. 1 Skenario Pengujian

No.	Komponen Yang Diuji	Skenario Pengujian	Jenis Pengujian
1.	*Main Menu*	Klik *Start Game*	*Black Box*
		Klik *How To Play*	*Black Box*
		Klik *Credits*	*Black Box*
		Klik *Exit*	*Black Box*

2.	*Scene Video*	Tekan Tombol Spasi	*Black Box*
3.	Memainkan Permainan	Tekan Tombol ESC	*Black Box*
		Klik *Continue*	*Black Box*
		Klik *Restart Level*	*Black Box*
		Klik *Main Menu*	*Black Box*
		Klik *Mission*	*Black Box*
		Klik *How To Play*	*Black Box*
		Klik *Credits*	*Black Box*
		Klik *Exit*	*Black Box*
		Berhasil Menyelesaikan Misi	*Black Box*
		Gagal Menyelesaikan Misi	*Black Box*
4.	*How To Play*	Klik *Back*	*Black Box*
5.	*Credits*	Klik *Back*	*Black Box*
6.	*Mission*	Klik *Back*	*Black Box*
7.	*Game Over*	Klik *Back*	*Black Box*
8.	*Mission Complete*	Klik *Start Menu*	*Black Box*
9.	Algoritma A*	Pencarian rute atau jalan terbaik untuk NPC	*White Box*
10.	Algoritma *Fuzzy Logic*	Perilaku NPC	*White Box*

IV.2.1.2 Kasus dan Hasil Pengujian (*Black Box Testing)*

Kasus dan hasil uji perangkat lunak yang sudah dibangun dengan menggunakan metode *black box* berdasarkan pengamatan yang dilakukan dan menghasilkan kesimpulan dari pengujian tersebut yang dijelaskan diantaranya sebagai berikut.

1. Pengujian *Main Menu*

Pengujian *main menu* dengan kasus dan hasil uji, kemudian diamati dan menghasilkan kesimpulan dapat dilihat pada tabel IV.2

Tabel IV. 2 Pengujian *Main Menu*

Kasus dan hasil uji (data normal)			
Masukan	**Yang diharapkan**	**Pengamatan**	**Kesimpulan**
Klik tombol *start game*	Memulai permainan dengan menampilkan *scene video* terlebih dahulu	Tombol dapat berfungsi sesuai dengan yang diharapkan	Diterima
Klik tombol *how to play*	Menampilkan halaman *how to play*	Tombol dapat berfungsi sesuai dengan yang diharapkan	Diterima
Klik tombol *credits*	Menampilkan halaman *credits*	Tombol dapat berfungsi sesuai dengan yang diharapkan	Diterima
Klik tombol *exit*	Keluar dari aplikasi *game*	Tombol dapat berfungsi sesuai dengan yang diharapkan	Diterima

2. Pengujian *Scene Video*

Pengujian *scene video* dengan kasus dan hasil uji, kemudian diamati dan menghasilkan kesimpulan dapat dilihat pada tabel IV.3

Tabel IV. 3 Pengujian *Scene Video*

Kasus dan hasil uji (data normal)			
Masukan	**Yang diharapkan**	**Pengamatan**	**Kesimpulan**
Tekan tombol spasi	Menghentikan *scene video* dan masuk ke dalam *level* permainan	Tombol dapat berfungsi sesuai dengan yang diharapkan	Diterima

3. Pengujian Memainkan Permainan

Pengujian memainkan permainan dengan kasus dan hasil uji, kemudian diamati dan menghasilkan kesimpulan dapat dilihat pada tabel IV.4

Tabel IV. 4 Pengujian Memainkan Permainan

Kasus dan hasil uji (data normal)			
Masukan	**Yang diharapkan**	**Pengamatan**	**Kesimpulan**
Tekan tombol ESC	Memberhentikan permainan dan menampilkan *pause menu*	Tombol dapat berfungsi sesuai dengan yang diharapkan	Diterima
Klik tombol *continue*	Menampilkan kembali halaman permainan	Tombol dapat berfungsi sesuai dengan yang diharapkan	Diterima
Klik tombol *restart level*	Mereset ulang halaman permainan kembali seperti pertama kali dimainkan	Tombol dapat berfungsi sesuai dengan yang diharapkan	Diterima
Klik tombol *main menu*	Keluar dari permainan dan menampilkan halaman *main menu*	Tombol dapat berfungsi sesuai dengan yang diharapkan	Diterima
Klik tombol *mission*	Menampilkan halaman *mission*	Tombol dapat berfungsi sesuai dengan yang diharapkan	Diterima
Klik tombol *how to play*	Menampilkan halaman *how to play*	Tombol dapat berfungsi sesuai dengan yang diharapkan	Diterima
Klik tombol *main credits*	Menampilkan halaman *credits*	Tombol dapat berfungsi sesuai dengan yang diharapkan	Diterima

Klik tombol *exit*	Keluar dari aplikasi *game*	Tombol dapat berfungsi sesuai dengan yang diharapkan	Diterima
Berhasil menyelesaikan misi	Menampilkan halaman *mission complete*	Tombol dapat berfungsi sesuai dengan yang diharapkan	Diterima
Gagal menyelesaikan misi	Menampilkan halaman *game over*	Tombol dapat berfungsi sesuai dengan yang diharapkan	Diterima

4. Pengujian *How To Play*

Pengujian *how to play* dengan kasus dan hasil uji, kemudian diamati dan menghasilkan kesimpulan dapat dilihat pada tabel IV.5

Tabel IV. 5 Pengujian *How To Play*

Kasus dan hasil uji (data normal)			
Masukan	**Yang diharapkan**	**Pengamatan**	**Kesimpulan**
Klik tombol *back*	Menampilkan kembali halaman sebelumnya	Tombol dapat berfungsi sesuai dengan yang diharapkan	Diterima

5. Pengujian *Credits*

Pengujian *credits* dengan kasus dan hasil uji, kemudian diamati dan menghasilkan kesimpulan dapat dilihat pada tabel IV.6

Tabel IV. 6 Pengujian *Credits*

Kasus dan hasil uji (data normal)			
Masukan	**Yang diharapkan**	**Pengamatan**	**Kesimpulan**
Klik tombol *back*	Menampilkan kembali halaman sebelumnya	Tombol dapat berfungsi sesuai dengan yang diharapkan	Diterima

6. Pengujian *Mission*

Pengujian *mission* dengan kasus dan hasil uji, kemudian diamati dan menghasilkan kesimpulan dapat dilihat pada tabel IV.7

Tabel IV. 7 Pengujian *Mission*

Kasus dan hasil uji (data normal)			
Masukan	**Yang diharapkan**	**Pengamatan**	**Kesimpulan**
Klik tombol *back*	Menampilkan kembali halaman sebelumnya	Tombol dapat berfungsi sesuai dengan yang diharapkan	Diterima

7. Pengujian *Game Over*

Pengujian *game over* dengan kasus dan hasil uji, kemudian diamati dan menghasilkan kesimpulan dapat dilihat pada tabel IV.8

Tabel IV. 8 Pengujian *Game Over*

Kasus dan hasil uji (data normal)			
Masukan	**Yang diharapkan**	**Pengamatan**	**Kesimpulan**
Klik tombol *back*	Menampilkan kembali halaman *main menu*	Tombol dapat berfungsi sesuai dengan yang diharapkan	Diterima

8. Pengujian *Mission Complete*

Pengujian *mission complete* dengan kasus dan hasil uji, kemudian diamati dan menghasilkan kesimpulan dapat dilihat pada tabel IV.9

Tabel IV. 9 Pengujian *Mission Complete*

Kasus dan hasil uji (data normal)			
Masukan	**Yang diharapkan**	**Pengamatan**	**Kesimpulan**
Klik tombol	Menampilkan kembali	Tombol dapat berfungsi	Diterima

start menu	halaman *main menu*	sesuai dengan yang diharapkan	

IV.2.1.3 Kasus dan Hasil Pengujian (*White Box Testing)*

1. Pengujian *White Box* Algoritma A*

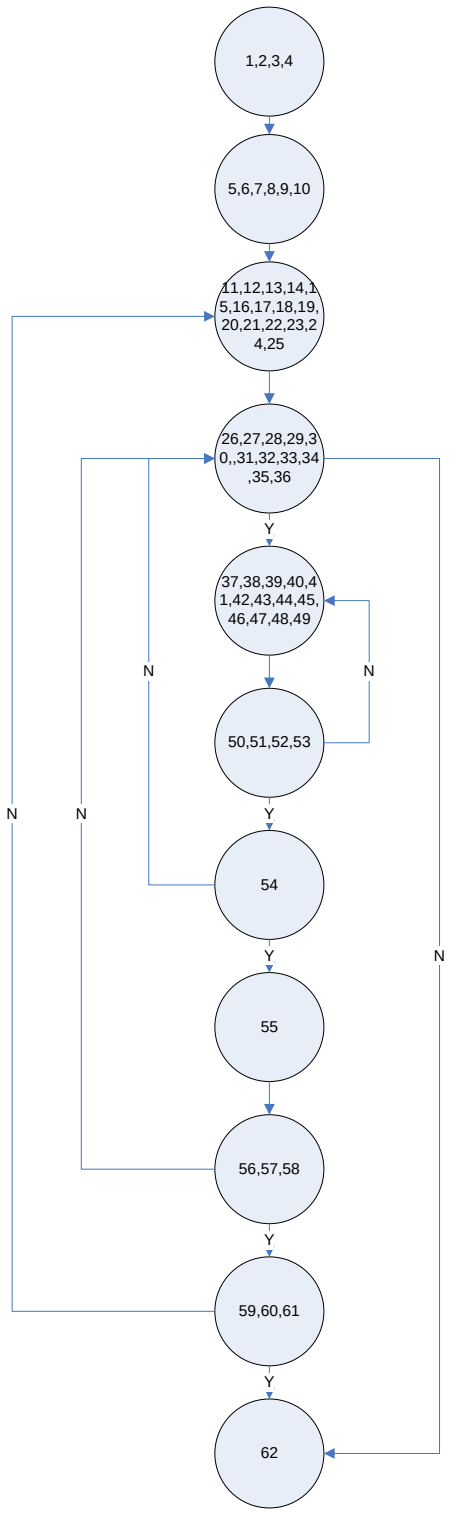
Berikut ini merupakan algoritma A* untuk *level* 3 yang penerapannya menggunakan bahasa pemrograman C#:

```
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 IV. 19 *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

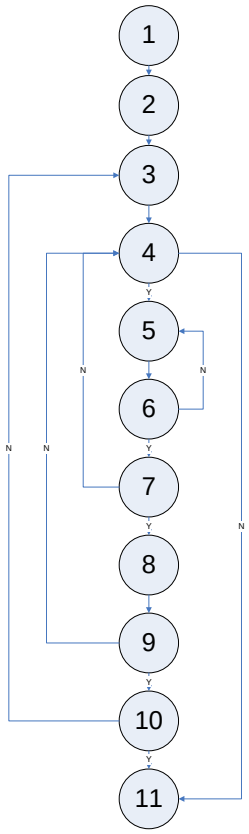
Path 4 = 1-2-3-4-5-6-7-8-9-10-11-12-13-14-15-16-17-18-19-20-21-22-23-24-25-26-27-28-29-30-31-32-33-34-35-36-37-38-39-40-41-42-43-44-45-46-47-48-49-50-51-52-53-54-26-27-28-29-30-31-32-33-34-35-36-37-38-39-40-41-42-43-44-45-46-47-48-49-50-51-52-53-54-55-56-57-58-59-60-61-62

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 IV. 20 Penyederhanaan *Flow Graph* Algoritma A*

Tabel IV. 10 *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
2. Nilai 1 merepresentasikan adanya keterhubungan antar simpul

e. Hasil Uji Algoritma A*

Tabel IV. 11 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].walkabl	Rute = null, rute tidak ada	Rute tidak ditemukan	[√] Alur terlewati [ ] Alur tidak terlewati

No	Kasus uji	Yang diharapkan	Pengamatan	Keterangan
	e && connections[i].area != area - doOpen			
2	- readerStream != null - writerStream != null - anchors != null - doOpen - connectionCosts == null - int i=0; i<connectionCosts.Leng th; i++	Current Node = Goal. Rute ditemukan	Rute ditemukan	[√] Alur terlewati [ ] Alur tidak terlewati
3	- readerStream != null - writerStream != null - anchors != null - doOpen - connectionCosts == null - int i=0; i<connectionCosts.Leng th; i++ - !doOpen - int i=0;i<connectionCosts. Length;i++ - other.connections != null - int q = 0;q < other.connections.Leng th;q++ - other.connections[q]	Node tidak dapat dilalui, pengujian collision atau unwalkable node	Node menjadi unwalkable	[√] Alur terlewati [ ] Alur tidak terlewati

No	Kasus uji	Yang diharapkan	Pengamatan	Keterangan
	== 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++			
4	- 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++	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
	- 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	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
	- connections[i].walkable && connections[i].area != area - 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			
6	- readerStream != null - writerStream != null - anchors != null - doOpen - connectionCosts == null - int i=0;	Open = null, tidak ada *node* di dalam open	Tidak ada *node* di dalam open	[√] Alur terlewati [ ] Alur tidak terlewati

No	Kasus uji	Yang diharapkan	Pengamatan	Keterangan
	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			

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.

2. Pengujian *White Box* Algoritma *Fuzzy Logic*

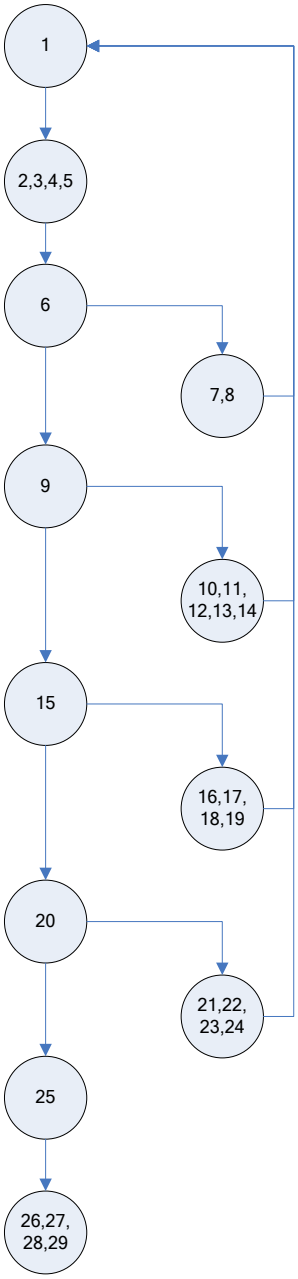
Berikut ini merupakan algoritma *fuzzy logic* untuk semua *level* yang penerapannya menggunakan bahasa pemrograman C#:

1.
```
jarak = (target.position - myTransform.position).magnitude;
```

```
2.  rules rule = new rules(jarak,hitPoints);
3.  defuzzyfikasi def = new defuzzyfikasi();
4.  rule.periksaRule();
5.  def. hasilDefuzzy();
6.  if((balikKeMarkas.position -
    myTransform.position).magnitude < 1.0){
7.  StartCoroutine(tambahDarah());
8.  }
9.  else if(def.hasilDefuzzyfikasi <= 10.0f) {
10. balikKeMarkas();
11. myTransform.position += myTransform.forward * moveSpeed *
    Time.deltaTime;
12. anim.animation.CrossFade(walkAnim);
13. playWalkSounds();
14. }
15. else if(def.hasilDefuzzyfikasi > 10.0f &&
    def.hasilDefuzzyfikasi <= 20.0f){
16. bertahan();
17. anim.animation.CrossFade(attackAnim2);
18. audio.Stop();
19. }
20. else if(def.hasilDefuzzyfikasi > 20.0f &&
    def.hasilDefuzzyfikasi <= 30.0f){
21. menyerang();
22. damage = Random.Range(20, 30);
23. anim.animation.CrossFade(walkAnim);
24. }
25. else if(def.hasilDefuzzyfikasi > 30.0f &&
    def.hasilDefuzzyfikasi <= 40.0f){
26. menyerang();
27. moveSpeed = 4;
28. damage = 30;
29. }
```

a. *Flow Graph* Algoritma *Fuzzy Logic*

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



Gambar IV. 21 *Flow Graph* Algoritma *Fuzzy Logic*

Keterangan:

= Menggambarkan kondisi

= Menggambarkan aksi

b. *Cyclomatic Complexity* V(G)

E = 15, N = 12

V(G) = E – N + 2

V(G) = 15 – 12 + 2

V(G) = 3 + 2 = 5

Keterangan

E = Jumlah aksi

N = Jumlah Kondisi

c. *Independent Path*

Path 1 = 1-2-3-4-5-6-7-9-10-11-12-13-15-16-17-18-20-21-22-23-25-26-27-28-29

Path 2 = 1-2-3-4-5-6-7-8-29

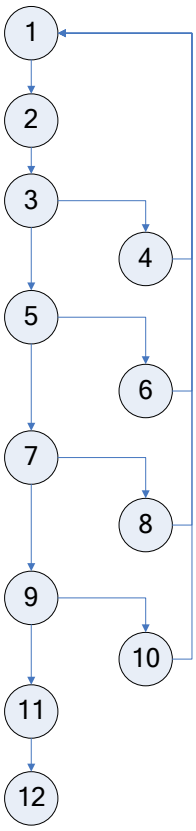
Path 3 = 1-2-3-4-5-6-7-9-10-11-12-13-14-29

Path 4 = 1-2-3-4-5-6-7-9-10-11-12-13-15-16-17-18-19-29

Path 5 = 1-2-3-4-5-6-7-9-10-11-12-13-15-16-17-18-20-21-22-23-24-29

d. *Graph Matriks* Algoritma *Fuzzy Logic*

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



Gambar IV. 22 Penyederhanaan *Flow Graph* Algoritma *Fuzzy Logic*

Tabel IV. 12 *Graph Matriks* Algoritma *Fuzzy Logic*

N	1	2	3	4	5	6	7	8	9	10	11	12	N(E) -1
1		1											1-1=0
2			1										1-1=0
3				1	1								2-1=1
4												1	1-1=0
5						1	1						2-1=1
6												1	1-1=0
7								1	1				2-1=1
8												1	1-1=0

9										1	1		2-1=1
10												1	1-1=0
11												1	1-1=0
12													1-1=0
												Sum(E)+1	4+1=5

Keterangan:

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

e. Hasil Uji Algoritma *Fuzzy Logic*

Tabel IV. 13 Hasil Uji Algoritma *Fuzzy Logic*

No	Kasus uji	Yang diharapkan	Pengamatan	Keterangan
1	- `balikKeMarkas.position myTransform.position.magnitude < 1.0` - `def.hasilDefuzzyfikasi <= 10.0f` - `def.hasilDefuzzyfikasi > 10.0f && def.hasilDefuzzyfikasi <= 20.0f` - `def.hasilDefuzzyfikasi > 20.0f && def.hasilDefuzzyfikasi <= 30.0f` - `def.hasilDefuzzyfikasi > 30.0f && def.hasilDefuzzyfikasi <= 40.0f`	Hasil defuzzy > 30 dan <= 40 NPC menyerang brutal	NPC menyerang brutal	[√] Alur terlewati [ ] Alur tidak terlewati

No	Kasus uji	Yang diharapkan	Pengamatan	Keterangan
2	- `balikKeMarkas.position myTransform.position.magnitude < 1.0`	NPC menambah *hitpoints* saat kembali ke markas	NPC menambah *hitpoints*	[√] Alur terlewati [ ] Alur tidak terlewati
3	- `balikKeMarkas.position myTransform.position.magnitude < 1.0` - `def.hasilDefuzzyfikasi <= 10.0f`	Hasil defuzzy <= 10 NPC melarikan diri ke markas	NPC melarikan diri ke markas	[√] Alur terlewati [ ] Alur tidak terlewati
4	- `balikKeMarkas.position myTransform.position.magnitude < 1.0` - `def.hasilDefuzzyfikasi <= 10.0f` - `def.hasilDefuzzyfikasi > 10.0f && def.hasilDefuzzyfikasi <= 20.0f`	Hasil defuzzy > 10 dan <= 20 NPC bertahan	NPC bertahan	[√] Alur terlewati [ ] Alur tidak terlewati
5	- `balikKeMarkas.position myTransform.position.magnitude < 1.0` - `def.hasilDefuzzyfikasi <= 10.0f` - `def.hasilDefuzzyfikasi > 10.0f && def.hasilDefuzzyfikasi <= 20.0f` - `def.hasilDefuzzyfikasi > 20.0f &&`	Hasil defuzzy > 20 dan <= 30 NPC menyerang	NPC menyerang	[√] Alur terlewati [ ] Alur tidak terlewati

No	Kasus uji	Yang diharapkan	Pengamatan	Keterangan
	`def.hasilDefuzzyfikasi <= 30.0f`			

f. Kesimpulan

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

IV.2.1.4 Kesimpulan Pengujian *Alpha*

Dari hasil pengujian yang dilakukan, dapat disimpulkan bahwa sistem sudah berjalan seperti yang diharapkan dan secara fungsional sudah dapat menghasilkan keluaran yang diharapkan, akan tetapi pada prosesnya masih memungkinkan untuk terjadinya kesalahan, namun frekuensi kesalahan masih relatif kecil.

IV.2.2 Pengujian *Beta*

Pengujian *beta* merupakan pengujian yang dilakukan secara objektif, dimana pengujian langsung melibatkan para pengguna aplikasi permainan secara langsung dengan menggunakan kuisioner mengenai kepuasan para pengguna terhadap aplikasi *game* yang dibangun dengan kandungan *point* syarat.

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

$$\mathbf{P=\Sigma s/skor\ ideal*100\%} \qquad \text{(IV.1)}$$

Keterangan:

P = Nilai presentase yang dicari

Σs = Jumlah skor yang didapatkan dari setiap pilihan jawaban dikalikan dengan nilai yang telah ditetapkan sebelumnya

Skor ideal = Nilai tertinggi pada *point* pilihan dikalikan dengan jumlah responden

Berikut ini adalah hasil persentase masing-masing jawaban yang sudah dihitung nilainya dengan menggunakan persamaan IV.1 diatas. Kuesioner ini telah diujikan kepada 10 orang. Hasil perhitungan persentase masing-masing jawaban adalah sebagai berikut.

Untuk pertanyaan No 1

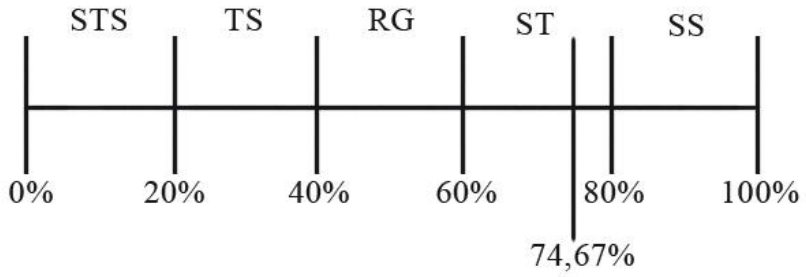
"Apakah anda setuju jika alur cerita dari *game The Walker* yang mengangkat sisi *humanism* sudah sesuai dengan alur cerita dari serial TV *The Walking Dead* "

Tabel IV. 14 Hasil Pengujian *Beta* Pertanyaan No 1

Kategori Jawaban	**Skor**	**Responden**	**S**
Sangat Setuju	5	0	0
Setuju	4	22	88
Ragu-ragu	3	8	24
Tidak Setuju	2	0	0
Sangat Tidak Setuju	1	0	0
JUMLAH		**30**	**112**

P = 112/150 * 100% = 74,67%

Berikut kriteria interpretasi skor:



Gambar IV. 23 Interpretasi Skor Pertanyaan No 1

Untuk pertanyaan No 2

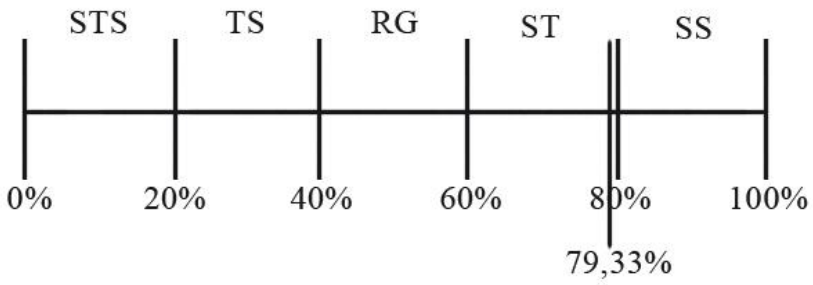
"Apakah desain grafis 3D di dalam *game The Walker* sudah menarik"

Tabel IV. 15 Hasil Pengujian *Beta* Pertanyaan No 2

Kategori Jawaban	Skor	Responden	S
Sangat Setuju	5	4	20
Setuju	4	21	84
Ragu-ragu	3	5	15
Tidak Setuju	2	0	0
Sangat Tidak Setuju	1	0	0
JUMLAH		**30**	**119**

P = 119/150 * 100% = 79,33%

Berikut kriteria interpretasi skor:



Gambar IV. 24 Interpretasi Skor Pertanyaan No 2

Untuk pertanyaan No 3

"Apakah anda merasa tertantang menyelesaikan *game The Walker* sampai *level* terakhir"

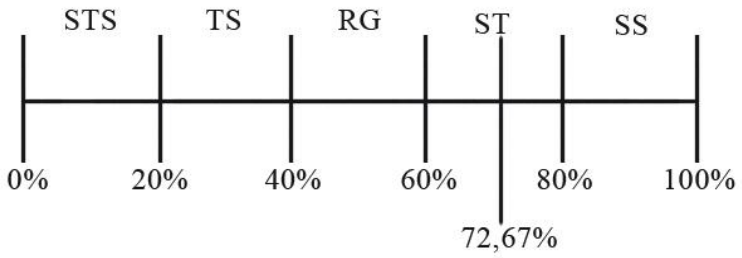
Tabel IV. 16 Hasil Pengujian *Beta* Pertanyaan No 3

Kategori Jawaban	Skor	Responden	S
Sangat Setuju	5	0	0
Setuju	4	19	76
Ragu-ragu	3	11	33

Tidak Setuju	2	0	0
Sangat Tidak Setuju	1	0	0
JUMLAH		**30**	**109**

P = 109/150 * 100% = 72,67%

Berikut kriteria interpretasi skor:



Gambar IV. 25 Interpretasi Skor Pertanyaan No 3

Untuk pertanyaan No 4

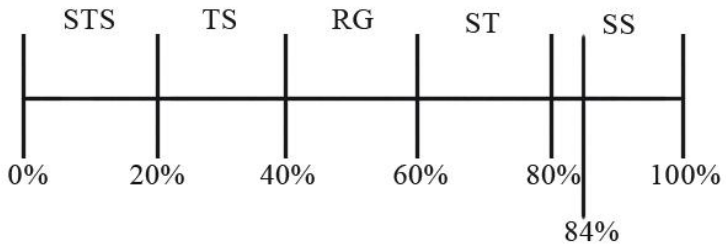
"Apakah *audio* seperti *sound effect* dan *music* yang digunakan sesuai untuk *game The Walker*"

Tabel IV. 17 Hasil Pengujian *Beta* Pertanyaan No 4

Kategori Jawaban	**Skor**	**Responden**	**S**
Sangat Setuju	5	11	55
Setuju	4	14	56
Ragu-ragu	3	5	15
Tidak Setuju	2	0	0
Sangat Tidak Setuju	1	0	0
JUMLAH		**30**	**126**

P = 0/150 * 100% = 84%

Berikut kriteria interpretasi skor:



Gambar IV. 26 Interpretasi Skor Pertanyaan No 4

Untuk pertanyaan No 5

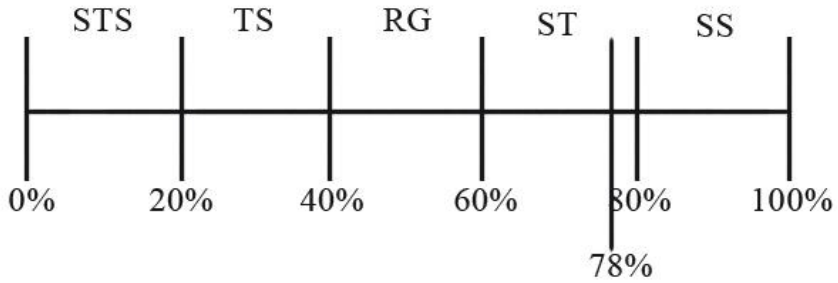
"Apakah informasi petunjuk di setiap *level* sudah membantu anda dalam menyelesaikan *game The Walker*"

Tabel IV. 18 Hasil Pengujian *Beta* Pertanyaan No 5

Kategori Jawaban	**Skor**	**Responden**	**S**
Sangat Setuju	5	4	20
Setuju	4	19	76
Ragu-ragu	3	7	21
Tidak Setuju	2	0	0
Sangat Tidak Setuju	1	0	0
JUMLAH		**30**	**117**

P = 117/150 * 100% = 78%

Berikut kriteria interpretasi skor:



Gambar IV. 27 Interpretasi Skor Pertanyaan No 5

Untuk pertanyaan No 6

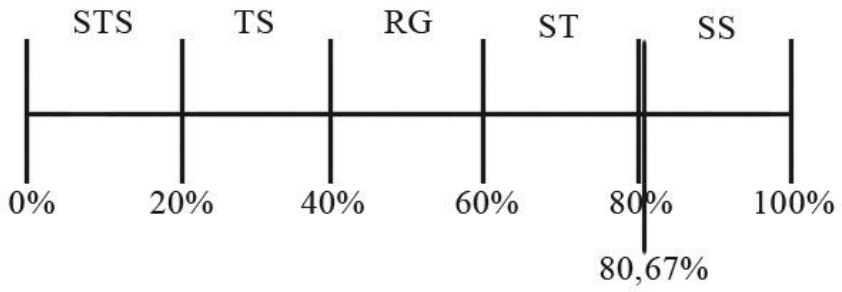
“Apakah musuh di dalam *game The Walker* sulit untuk dikalahkan”

Tabel IV. 19 Hasil Pengujian *Beta* Pertanyaan No 6

Kategori Jawaban	**Skor**	**Responden**	**S**
Sangat Setuju	5	6	30
Setuju	4	19	76
Ragu-ragu	3	5	15
Tidak Setuju	2	0	0
Sangat Tidak Setuju	1	0	0
JUMLAH		**30**	**121**

P = 121/150 * 100% = 80,67%

Berikut kriteria interpretasi skor:



Gambar IV. 28 Interpretasi Skor Pertanyaan No 6

Untuk pertanyaan No 7

“Apakah anda setuju *game The Walker* menarik untuk dimainkan”

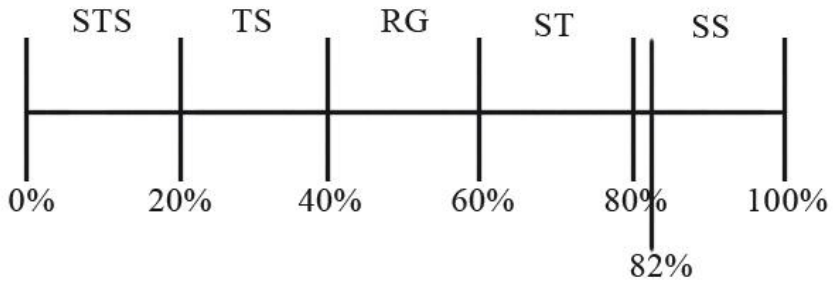
Tabel IV. 20 Hasil Pengujian *Beta* Pertanyaan No 7

Kategori Jawaban	**Skor**	**Responden**	**S**
Sangat Setuju	5	5	25
Setuju	4	23	92
Ragu-ragu	3	2	6

Tidak Setuju	2	0	0
Sangat Tidak Setuju	1	0	0
JUMLAH		**30**	**123**

P = 123/150 * 100% = 82%

Berikut kriteria interpretasi skor:



Gambar IV. 29 Interpretasi Skor Pertanyaan No 7

IV.2.2.1 Kesimpulan Pengujian *Beta*

Berdasarkan dari jawaban yang diberikan oleh responden, bisa ditarik kesimpulan bahwa "*Game Action* 3D *The Walker*" dinyatakan sudah sesuai dengan alur cerita serial TV *The Walking Dead*, memiliki desain grafis 3D yang menarik, memiliki tantangan untuk menyelesaikan *game* sampai *level* terakhir, *audio* yang digunakan sudah sesuai, memiliki bantuan informasi petunjuk di setiap *level*, dengan musuh yang sulit untuk dikalahkan, dan aplikasi *game The Walker* menarik untuk dimainkan.