

# **RANCANG BANGUN GAME MAZE 2D “RETURN TO EARTH”**

## **SKRIPSI**

Digunakan Sebagai Syarat Maju Ujian Diploma IV  
Politeknik Negeri Malang

Oleh :

**ARI MAHARDIKA AHMAD NAFIS**

**NIM. 1341180068**



**PROGRAM STUDI TEKNIK INFORMATIKA  
JURUSAN TEKNOLOGI INFORMASI  
POLITEKNIK NEGERI MALANG  
JUNI 2017**

**HALAMAN PENGESAHAN**  
**RANCANG BANGUN GAME MAZE 2D “RETURN TO**  
**EARTH”**

**Disusun Oleh :**  
**ARI MAHARDIKA AHMAD NAFIS      NIM.1341180068**

**Skripsi ini telah diuji pada tanggal 8 Juni 2017**

**Disetujui oleh :**

1. Penguji I	: <u>Erfan Rohadi, ST., M.Eng</u>	
	NIP. 19720123 200801 1 006	.....
2. Penguji II	: <u>Arief Prasetyo, S.Kom</u>	
	NIP. 19790313 200812 1 002	.....
3. Pembimbing I	: <u>Ridwan Rismanto, S.ST., M.KOM</u>	
	NIP. 19860318 201212 1 001	.....
4. Pembimbing II	: <u>Dyah Ayu Irawati, ST., M.CS</u>	
	NIP. 19840708 200812 2 001	.....

Mengetahui,

Ketua Jurusan  
Teknologi Informasi

Ketua Program Studi  
Teknik Informatika

Rudy Ariyanto, S.T., M.Cs  
NIP. 19711110 199903 1 002

Ir. Deddy Kusbianto P., M.MKom.  
NIP. 19621128 198811 1 001

## **HALAMAN PERNYATAAN**

### **PERNYATAAN**

Dengan ini saya menyatakan bahwa Skripsi ini tidak terdapat karya yang pernah diajukan untuk memperoleh gelar kesarjanaan di suatu Perguruan Tinggi, dan sepanjang pengetahuan saya juga tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis diacu dalam naskah ini dan disebutkan dalam daftar pustaka.

Batu, 1 Juni 2017

Ari Mahardika Ahmad Nafis

## ABSTRAK

**Nafis, Ari Mahardika Ahmad.** “Rancang Bangun *Game Maze 2D “Return To Earth”*”. Pembimbing: (1) **Ridwan Rismanto., S.ST., M.Kom,** (2) **Dyah Ayu Irawati, ST., M.Cs.**  
**Skripsi, Program Studi Teknik Informatika, Jurusan Teknologi Informasi, Politeknik Negeri Malang. 2017**

Salah satu unsur dari *game* yang membuat suatu *game* menjadi menarik adalah keberadaan NPC (*non-playable character*) pada *game* yang bisa menjadi lawan main player. Untuk bisa membuat NPC bergerak dan mengambil keputusan seperti manusia diperlukan kecerdasan buatan yang diimplementasikan pada NPC tersebut.

Penelitian ini akan dilakukan dengan menggunakan metode A\* yang diimplementasikan pada tingkah laku NPC untuk menemukan jalur menuju karakter player. Metode A\* (A-Star) merupakan metode untuk melakukan pencarian jalur terdekat dari titik awal ke titik tujuan. Selain itu dalam penelitian ini dikembangkan sebuah *game* 2D berjudul “*Return To Earth*” sebagai media untuk mengimplementasikan metode A\* dalam mengendalikan pergerakan NPC. Game ini menceritakan seorang astronot yang berusaha kembali ke Bumi setelah diculik oleh alien. Astronot harus berusaha menuju titik tujuan dengan menghindari alien yang mendekati karakter astronot.

Berdasarkan hasil uji coba yang dilakukan, diperoleh kesimpulan bahwa jalur yang diambil oleh algoritma A\* berhasil menemukan jalur terpendek untuk setiap level. Dengan demikian algoritma A\* ini dapat digunakan untuk mengambil jalur terdekat dari NPC menuju player.

**Kata Kunci :** *Video Game, Algoritma A\*, Puzzle*

## ***ABSTRACT***

***Nafis, Ari Mahardika Ahmad. “Rancang Bangun Game Maze 2D “Return To Earth””. Advisors: (1) Ridwan Rismanto., S.ST., M.Kom, (2) Dyah Ayu Irawati, ST., M.Cs.***

***Thesis, Informatic Engineering Study Program, Departement of Information Technology, State Polytechnic of Malang, 2017.***

*One of the points that make a game interesting is there is a NPC (non-playable character) in the game that could become a challenger for the player. To make NPC moves and makes decision like a human, it needs an artificial intelligence implemented into that NPC.*

*This research is conducted using A\* method that implemented into NPC behavior to find a route to player character. A\* method is a method to find the shortest route from the initial point to the target point. Moreover, in this research, a 2D game titled “Return To Earth” is developed as media for implementing A\* method to control NPC behavior. This game is about an astronaut trying to return to Earth after being kidnapped by an alien. The astronaut must try to reach the goal while avoiding the alien that moves closer to the astronaut character.*

*Based on the conducted test, it can be concluded that the route taken by A\* algorithm managed to find the shortest path for each level. Thus, A\* algorithm can be used to get the closest route from NPC to player.*

***Keywords : Video Game, A\* Algorithm, Puzzle***

## KATA PENGANTAR

Puji Syukur penulis panjatkan kehadiran Allah SWT atas segala rahmat dan hidayah-Nya penulis dapat menyelesaikan skripsi dengan judul “Rancang Bangun Game Maze 2D “Return To Earth””. Skripsi ini penulis susun sebagai persyaratan untuk menyelesaikan studi program Diploma IV Program Studi Teknik Informatika, Jurusan Teknologi Informasi, Politeknik Negeri Malang.

Penulis menyadari tanpa adanya dukungan dan kerja sama dari berbagai pihak, kegiatan skripsi ini tidak akan dapat berjalan baik. Untuk itu, penulis ingin menyampaikan rasa terima kasih kepada:

1. Bapak Rudy Ariyanto, ST., M.Cs, selaku ketua jurusan Teknologi Informasi.
2. Bapak Ir. Deddy Kusbianto P., M.Mkom., selaku ketua program studi Teknik Informatika.
3. Bapak Ridwan Rismanto, S.ST., M.Kom., dan Bu Dyah Ayu Irawati, ST., M.Cs, selaku pembimbing skripsi.
4. Keluarga dan teman-teman angkatan 2013 Teknik Informatika Politeknik Negeri Malang.
5. Dan seluruh pihak yang membantu dan mendukung lancarnya pembuatan Laporan Akhir dari awal hingga akhir yang tidak dapat saya sebutkan satu-persatu, Penulis menyadari bahwa dalam penyusunan laporan akhir ini, masih banyak terdapat kekurangan dan kelemahan yang dimiliki penulis baik itu sistematika penulisan maupun menggunakan bahasa. Untuk itu penulis mengharapkan saran dan kritik dari berbagai pihak yang bersifat membangun demi penyempurnaan laporan ini. Semoga laporan ini berguna bagi pembaca secara umum dan penulis secara khusus. Akhir kata, penulis ucapkan banyak terima kasih.

Batu, 1 Juni 2017

Penulis

## DAFTAR ISI

HALAMAN PENGESAHAN.....	ii
HALAMAN PERNYATAAN.....	iii
ABSTRAK.....	iv
<i>ABSTRACT</i> .....	v
KATA PENGANTAR.....	vi
DAFTAR ISI.....	vii
DAFTAR GAMBAR.....	ix
DAFTAR TABEL.....	x
DAFTAR LAMPIRAN.....	xi
BAB I PENDAHULUAN.....	1
1.1 Latar Belakang.....	1
1.2 Rumusan Masalah.....	2
1.3 Tujuan.....	2
1.4 Batasan Masalah.....	2
1.5 Sistematika Penulisan.....	3
BAB II. LANDASAN TEORI.....	4
2.1 Algoritma A*.....	4
2.2 C# (C-Sharp).....	5
2.3 Unity3D.....	6
2.4 Inkscape.....	6
2.5 AI ( <i>Artificial Intelligence</i> ).....	6
2.6 <i>Flowchart</i> .....	8
2.7 Storyboard.....	9
2.8 <i>Non Player Character</i> .....	9
2.9 Elemen dasar game.....	10
BAB III. METODOLOGI PENELITIAN.....	11
3.1 Metode Pengumpulan Data .....	11
3.2 Metodologi Pengembangan Sistem.....	11
BAB IV. ANALISIS DAN PERANCANGAN.....	12
4.1 Analisis.....	14
4.1.1 Gambaran Umum Sistem.....	14
4.1.2 Analisis Kebutuhan.....	14
4.2 Perancangan.....	15
4.2.1 Perancangan Karakter.....	15
4.2.2 Perancangan Gameplay.....	15
4.2.3 Perancangan Perilaku <i>Non-Playable Character</i> .....	17
4.2.4 Perancangan Kondisi Menang.....	17
4.2.5 Perancangan <i>Storyboard</i> .....	20
4.2.6 Perancangan Algoritma.....	22
BAB V. IMPLEMENTASI.....	24
5.1 Implementasi pada Inkscape.....	24
5.1.1 Pembuatan Logo <i>Game</i> .....	24
5.1.2 Pembuatan Karakter <i>Game</i> .....	25
5.1.3 Pembuatan Lingkungan <i>Game</i> .....	25
5.1.3.1 <i>Main Menu</i> .....	25
5.1.3.2 <i>Panel</i> .....	26

5.1.3.3 Tombol-Tombol.....	26
5.1.3.4 <i>Background</i> .....	27
5.1.4 Mengekspor Gambar Menjadi <i>Sprite</i> .....	27
5.2 Implementasi pada Unity.....	29
5.2.1 Main Menu.....	29
5.2.2 Level Picker.....	30
5.2.3 Desain Background <i>Gameboard</i> .....	30
5.2.4 Penentuan Jumlah Langkah Tiap Level.....	31
5.2.5 Implementasi Algoritma A*.....	32
BAB VI. PENGUJIAN DAN PEMBAHASAN.....	35
6.1 Pengujian.....	35
6.1.1 Pengujian <i>Alpha</i> .....	35
6.1.1.1 Kasus dan Hasil Pengujian <i>Alpha</i> .....	35
6.1.1.2 Kesimpulan Pengujian <i>Alpha</i> .....	37
6.1.2 Pengujian Beta.....	37
6.1.2.1 Kuesioner Pengujian Beta.....	37
6.1.2.2 Hasil Pengujian Beta.....	38
6.1.3 Pengujian Algoritma A*.....	39
6.2 Pembahasan.....	43
6.2.1 Tingkat Keberhasilan Algoritma A*.....	44
BAB VII. KESIMPULAN.....	45
7.1 Kesimpulan.....	45
7.2 Saran.....	45
DAFTAR PUSTAKA.....	46
LAMPIRAN-LAMPIRAN.....	48



## DAFTAR GAMBAR

Gambar 3.1 Tampilan Game <i>Mummy Maze Deluxe</i> .....	15
Gambar 4.1 Desain Karakter Game.....	15
Gambar 4.2 Desain Karakter NPC.....	15
Gambar 4.3 Flowchart Perilaku NPC.....	17
Gambar 4.4 Flowchart Kondisi Menang Level 1-6.....	18
Gambar 4.5 Flowchart Kondisi Menang Level 7-9.....	19
Gambar 4.6 Flowchart A*.....	22
Gambar 4.7 Flowchart A* (Lanjutan).....	23
Gambar 5.1 Logo Inkscape.....	24
Gambar 5.2 Pembuatan Logo Game.....	24
Gambar 5.3 Logo <i>Game</i> .....	25
Gambar 5.4 Pembuatan Karakter.....	25
Gambar 5.5 Desain gambar pada main menu.....	26
Gambar 5.6 Desain gambar panel.....	26
Gambar 5.7 Desain tombol untuk game.....	27
Gambar 5.8 Desain luar angkasa.....	27
Gambar 5.9 Mengekspor gambar background.png.....	28
Gambar 5.10 Gambar background.png.....	28
Gambar 5.11 Logo Unity.....	29
Gambar 5.12 Main Menu.....	29
Gambar 5.13 Scene Level Picker.....	30
Gambar 5.14 Desain Background Gameboard.....	31
Gambar 5.15 <i>Hierarchy Game "Return To Earth"</i> .....	32
Gambar 5.16 <i>Inspector World(Astar)</i> .....	33
Gambar 5.17 Fungsi pada script Pathfinding.....	34
Gambar 5.18 Fungsi pada script MoveNPC.....	34
Gambar 6.1 Pengujian Algoritma A*.....	39
Gambar 6.2 Pengujian Algoritma Menggunakan Perhitungan Manual.....	39

## DAFTAR TABEL

Tabel 2.1 Tabel Flowchart.....	8
Tabel 4.1 Pembeda Tiap Level.....	16
Tabel 4.2 Storyboard.....	20
Tabel 5.1 Penentuan Jumlah Langkah.....	30
Tabel 5.2 Langkah yang Disediakan untuk Tiap Level.....	30
Tabel 6.1 Tabel Kasus Uji.....	35
Tabel 6.2 Kuesioner Pengujian Beta.....	38
Tabel 6.3 Keterangan nilai kuesioner.....	38
Tabel 6.4 Hasil Kuesioner.....	38
Tabel 6.5 Pengujian Algoritma A* .....	39

## **DAFTAR LAMPIRAN**

Lampiran 1	Lembar Bimbingan Dosen Pembimbing 1.....	48
Lampiran 2	Lembar Bimbingan Dosen Pembimbing 2.....	49
Lampiran 3	Listing Program.....	50
Lampiran 4	Kuesioner.....	59
Lampiran 5	Profil Penulis.....	69

# BAB I PENDAHULUAN

## 1.1 Latar Belakang

Sampai saat ini peminat industri *game* tidak pernah surut, bahkan semakin meningkat dari waktu ke waktu. Anak-anak, remaja, dewasa, bahkan orangtua juga banyak yang gemar bermain *game* di *handphone*, *tablet*, atau laptop. Salah satu unsur dari *game* yang membuat suatu *game* menjadi menarik adalah keberadaan AI atau *artificial intelligence* (kecerdasan buatan) pada NPC atau *non-playable character* pada *game* yang bisa menjadi lawan main *player*. Kecerdasan buatan adalah salah satu bidang dalam ilmu komputer yang membuat komputer agar bertindak dan sebaik seperti manusia (menirukan kerja otak manusia).

Salah satu tantangan penerapan AI dalam *game* adalah bagaimana mengimplementasikan algoritma kecerdasan buatan pada *game* yang dibentuk *maze* atau labirin. Pada *game* yang akan dikembangkan, *player* akan berusaha untuk keluar dari *maze* dengan terus menghindari NPC yang mengejar *character player*, karena NPC pada *game* ini harus bisa mengejar *player*, maka diperlukan algoritma yang bisa membuat NPC bisa mengejar *character player*. Pada *game* yang akan dikembangkan ini, penulis menggunakan algoritma A\* yang akan diimplementasikan pada NPC.

Metode A\* dikembangkan oleh Peter Hart, Nils Nilsson dan Bertram Raphael. Metode A\* adalah sebuah *graph* atau metode pohon pencarian yang digunakan untuk mencari jalan dari sebuah *node* awal ke *node* tujuan (*goal node*) yang telah ditentukan. Metode ini menggunakan “estimasi heuristic”  $h(n)$  pada setiap *node* untuk mengurutkan setiap *node*  $n$  berdasarkan estimasi rute terbaik yang melalui *node* tersebut. Algoritma A\* mencari jalur dengan *cost* terkecil dari *node* awal ke *node* tujuan [1].

Algoritma A\* sering digunakan dalam *game*, seperti dalam penelitian yang dilakukan oleh Agung Pamungkas, Eka Puji Widiyanto, dan Renni Anggreni dengan judul “Penerapan Algoritma A\* pada *Game* Edukasi *The Maze Island* Berbasis *Android*” pada tahun 2011. Algoritma A\* juga diimplementasikan pada bidang lain selain *game*, seperti pada penelitian sebelumnya yang telah dilakukan

oleh Muhammad Irsyad dan Endang Rasilla dengan judul "Aplikasi Pencarian Lokasi Gedung dan Ruangan Universitas Negeri Sultan Syarif Kasim Riau pada Platform Android Menggunakan Algoritma A-Star (A\*)" pada tahun 2015, dapat diketahui bahwa metode A\* digunakan untuk pencarian jalur terpendek. Dan didapatkan hasil bahwa Algoritma A-Star memberikan informasi rute terdekat yang akurat dengan 10 kali percobaan dan menghasilkan rute yang sama antara pencarian rute secara manual dengan pencarian rute pada aplikasi. [2]

Penelitian juga dilakukan oleh Mulia Purwati, Okti Firnawati, dan Willy, dengan judul "Penerapan Algoritme A\* (*A STAR*) Dalam Optimasi Penentuan Halte Transmisi di Palembang Berbasis Android", dan didapatkan hasil yang sesuai, yang berarti hasilnya sesuai dengan perhitungan manual berdasarkan parameter yang digunakan.

Berdasarkan latar belakang diatas, penulis berharap algoritma A\* ini dapat meningkatkan unsur keatraktifan pada *game* yaitu dengan mengimplementasikannya pada NPC.

## 1.2 Rumusan Masalah

Berdasarkan uraian latar belakang yang telah dijelaskan sebelumnya, rumusan masalah dalam skripsi ini adalah :

- a) Bagaimana membangun *Game* 2D Puzzle "*Return to Earth*" dengan mengimplementasikan metode A\* pada AI lawan.
- b) Bagaimana merancang map agar bisa mengimplementasikan algoritma A\*.

## 1.3 Tujuan

Tujuan dari pengembangan *game* 2D Puzzle "*Return to Earth*" ini adalah :

- a) Dapat mengimplementasikan algoritma A\* ke AI didalam *game*.
- b) Merancang map yang sesuai agar bisa mengimplementasikan algoritma A\*.

## 1.4 Batasan Masalah

Ada beberapa batasan masalah dalam pembuatan skripsi ini agar pembahasan lebih terfokus sesuai dengan tujuan yang akan dicapai. Batasan masalah dari *Game* 2D Puzzle "*Return to Earth*" ini adalah :

- a) *Game* bersifat *offline* 2D, tidak terhubung dengan jaringan internet.

- b) *Game* hanya dapat dimainkan oleh satu pemain atau *single player*.
- c) Menggunakan algoritma A\* pada AI lawan.
- d) *Software* pembangun *game* ini menggunakan Unity 5.50f3.
- e) *Software* desain untuk *game* ini menggunakan Inkscape 0.91.
- f) Bahasa pemrograman yang digunakan adalah C#.

## **1.5 Sistematika Penulisan**

### **BAB I PENDAHULUAN**

Bab ini berisi penjelasan mengenai latar belakang mengapa melakukan penelitian ini, rumusan masalah, tujuan, serta batasan masalah dari penelitian ini.

### **BAB II LANDASAN TEORI**

Bab ini berisi penjelasan mengenai algoritma, bahasa pemrograman, dan aplikasi yang digunakan untuk membangun *game* ini. Bab ini juga berisi penjelasan mengenai istilah-istilah asing yang digunakan dalam laporan ini.

### **BAB III METODOLOGI PENELITIAN**

Bab ini berisi penjelasan mengenai cara pengumpulan data yang digunakan serta metode yang digunakan dalam membangun *game* ini.

### **BAB IV ANALISIS DAN PERANCANGAN**

Bab ini berisi penjelasan mengenai analisa dan bagian-bagian yang perlu dipersiapkan terlebih dahulu sebelum melakukan implementasi, seperti *flowchart*, *storyboard*, serta desain.

### **BAB V IMPLEMENTASI**

Bab ini berisi penjelasan mengenai implementasi pada aplikasi yang digunakan dalam pembuatan *game* ini, seperti pembuatan *sprites* yang akan digunakan dalam *game* serta implementasi algoritma pada *game*.

### **BAB VI PENGUJIAN DAN PEMBAHASAN**

Bab ini berisi mengenai hasil pengujian dari *game* yang dikembangkan serta pembahasan dari algoritma yang digunakan.

### **BAB VII PENUTUP**

Bab ini berisi kesimpulan dari penelitian yang dilakukan serta saran yang bisa dilakukan apabila ingin mengembangkan penelitian ini.

## BAB II. LANDASAN TEORI

### 2.1 Algoritma A\*

Algoritma ini merupakan algoritma *Best First Search* yang menggabungkan *Uniform Cost Search* dan *Greedy Best-First Search*. Biaya yang diperhitungkan didapat dari biaya sebenarnya ditambah dengan biaya perkiraan. Dalam notasi matematika dituliskan sebagai :

$$f(n) = g(n) + h(n) \quad f(n) = g(n) + h(n) \quad (3.1)$$

Dengan  $g(n)$   $g(n)$  adalah *cost of the path* dari keadaan awal ke *node n* dan  $h(n)$   $h(n)$  adalah perkiraan heuristik atau *cost* atau *path* dari *node n* ke tujuan. Jadi,  $f(n)$   $f(n)$  adalah perkiraan total *cost* terendah dari setiap *path* yang akan dilalui *node n* ke *node* tujuan. Dengan kata lain, *cost* adalah jarak yang telah ditempuh, dan panjang garis lurus antara *node n* dengan *node* akhir adalah perkiraan heuristiknya [3]. Dengan perhitungan biaya seperti ini algoritma A\* adalah *complete* dan *optimal*.

Sama dengan algoritma dasar *Best First Search*, algoritma A\* ini juga menggunakan dua senarai: *OPEN* dan *CLOSED*. Terdapat tiga kondisi bagi setiap suksesor yang dibangkitkan, yaitu : sudah berada di *OPEN*, sudah berada di *CLOSED*, dan tidak berada di *OPEN* maupun *CLOSED*. Pada ketiga kondisi tersebut diberikan penanganan yang berbeda-beda.

Jika suksesor sudah pernah berada di *OPEN*, maka dilakukan pengecekan apakah perlu pengubahan *parent* atau tidak tergantung pada nilai  $g$ -nya melalui *parent* lama atau *parent* baru. Jika melalui *parent* baru memberikan nilai  $g$  yang lebih kecil, maka dilakukan pengubahan *parent*. Jika pengubahan *parent* dilakukan, maka dilakukan pula perbaruan (*update*) nilai  $g$  dan nilai  $f$  pada suksesor tersebut. Dengan perbaruan ini, suksesor tersebut memiliki kesempatan yang lebih besar untuk terpilih sebagai simpul terbaik (*best node*).

Jika suksesor sudah pernah berada di *CLOSED*, maka dilakukan pengecekan apakah perlu pengubahan *parent* atau tidak. Jika ya, maka dilakukan perbaruan nilai  $g$  dan  $f$  pada suksesor tersebut serta semua “anak cucunya” yang sudah berada di *OPEN*. Dengan perbaruan ini, maka semua anak cucunya tersebut memiliki kesempatan yang lebih besar untuk terpilih sebagai simpul terbaik (*best node*).

Jika suksesor tidak berada di *OPEN* maupun *CLOSED*, maka suksesor tersebut dimasukkan ke dalam *OPEN*. Tambahkan suksesor tersebut sebagai suksesornya *best node*. Hitung biaya suksesor tersebut dengan rumus  $f = g + h$  [4]

*Pseudocode* dari A\* adalah sebagai berikut :

```

initialize the open list
initialize the closed list
put the starting node on the open

while the open list is not empty
    find the node with the least  $f$  on the open, call it "current"
    remove current from open
    add current to closed

    if current is the target node, then path has been found
        Return

    foreach neighbour of the current node
        if neighbour is not traversable or neighbour is in closed
            skip to the next neighbour

        if new path to neighbour is shorter or neighbour is not in
open
            set  $f\_cost$  of neighbour
            set parent of neighbour to current
            if neighbour is not in open
                add neighbour to open

```

## 2.2 C# (C-Sharp)

C# adalah bahasa pemrograman komputer seperti C, C++, Java, maupun yang lainnya. Perbedaanannya, C# menggunakan *library* kelas yang terdapat pada .NET Framework. Hal ini tentu berbeda dengan C, C++, maupun Java, yang masing-masing memiliki *library* kelas sendiri-sendiri. Kelebihan *library* kelas yang terdapat di dalam .NET Framework adalah dapat digunakan oleh bahasa-bahasa lain yang mendukung .NET seperti Visual Basic (VB), dan Visual C++ (VC).[5]



### 2.3 Unity3D

Unity3D adalah mesin 3D yang sangat kuat dan *cross-platform*, serta lingkungan pengembangan yang *user-friendly*. Cukup mudah untuk pemula, dan cukup kuat untuk ahli. Unity pasti menarik perhatian siapapun yang ingin menciptakan game 3D dan aplikasi untuk *mobile*, *desktop*, *web*, dan *consoles* dengan mudah.[6]

### 2.4 Inkscape

Inkscape adalah *software* profesional untuk menggambar gambar vector dengan kualitas baik yang berjalan pada *Windows*, *Mac OSX*, dan *GNU/Linux*. *Software* ini digunakan oleh profesional dan penghobi diseluruh dunia, untuk menciptakan grafis dengan variasi yang sangat luas, seperti ilustrasi, ikon, logo, diagram, map, dan gambar *website*. Inkscape menggunakan standar terbuka W3C, SVG (*Scalable Vector Graphics*) sebagai format aslinya, dan gratis serta merupakan *software OPEN-source*.

Inkscape memiliki alat menggambar yang mutakhir dengan kemampuan yang bisa dibandingkan dengan Adobe Illustrator, CorelDRAW, dan Xara Xtreme. *Software* ini bisa melakukan impor dan ekspor ke berbagai format *file*, termasuk SVG, AI, EPS, PDF, PS, dan PNG. *Software* ini juga memiliki set fitur yang luas, tampilan yang sederhana, layanan multi-bahasa, dan didesain supaya bisa dikembangkan, pengguna bisa mengkustomisasi fungsionalitas Inkscape dengan *add-ons*.

Projek Inkscape memiliki komunitas pengguna internasional yang berkembang, dan terdapat banyak material belajar yang tersedia untuk membantu dengan kreasimu. Bantuan dan layanan juga disediakan oleh komunitas dan terdapat banyak jalur untuk bisa ikut bergabung jika ingin membantu mengembangkan projek Inkscape.[7]

### 2.5 AI (*Artificial Intelligence*)

*Artificial Intelligence* mengandung arti tiruan atau kecerdasan. Secara harfiah *Artificial Intelligence* adalah kecerdasan buatan. Kecerdasan buatan adalah

salah satu bidang dalam ilmu komputer yang membuat komputer agar bertindak dan sebaik seperti manusia (menirukan kerja otak manusia) [8].

Ada pula beberapa definisi AI yang disampaikan oleh beberapa ahli. Para ahli mendefinisikan AI secara berbeda-beda tergantung pada sudut pandang mereka masing-masing. Ada yang fokus pada logika berpikir manusia saja, tetapi ada juga yang mendefinisikan AI secara lebih luas pada tingkah laku manusia. Pada [RUS95], Stuart Russel dan Peter Norvig mengemompokkan definisi AI yang diperoleh dari beberapa *textbook* berbeda, ke empat kategori, yaitu:

#### 1. *Thinking humanly: the cognitive modelling approach*

Pendekatan ini dilakukan dengan dua cara sebagai berikut:

- Melalui intropeksi: mencoba menangkap pemikiran-pemikiran kita sendiri ada saat kita berpikir. Tetapi seorang psikolog Barat mengatakan: “*how do you know that you understand?*” Bagaimana Anda tahu bahwa Anda mengerti? Karena pada saat Anda menyadari pemikiran Anda, ternyata pemikiran tersebut sudah lewat dan digantikan dengan kesadaran Anda. Sehingga, definisi ini terkesan mengada-ada dan tidak mungkin dilakukan.
- Melalui eksperimen-eksperimen psikologi.

#### 2. *Acting humanly: the Turing test approach*

Pada tahun 1950, Alan Turing merancang suatu ujian bagi komputer berintelejensi untuk menguji apakah komputer tersebut mampu mengelabui seorang manusia yang menginterogasinya melalui *teletype* (komunikasi berbasis teks jarak jauh). Jika *interrogator* tidak dapat membedakan yang diinterogasi adalah manusia atau komputer, maka komputer berintelejensi tersebut lolos *Turing test*. Komputer tersebut perlu memiliki kemampuan *Natural Language Processing, Knowledge Representation, Automated Reasoning, Machine Learning, Computer Vision, Robotics*. *Turing test* sengaja menghindari interaksi fisik antara *interrogator* dan komputer karena simulasi fisik manusia tidak memerlukan intelegensi.

#### 3. *Thinking rationally: the laws of thought approach*

Terdapat dua masalah dalam pendekatan ini yaitu :

- Tidak mudah untuk membuat pengetahuan informal yang menyatakan pengetahuan tersebut ke dalam *formal term* yang diperlukan oleh notasi logika, khususnya ketika pengetahuan tersebut memiliki kepastian kurang dari 100%.
- Terdapat perbedaan besar antara dapat memecahkan masalah “dalam prinsip” dan memecahkannya “dalam dunia nyata”.

#### 4. *Acting rationally: the rational agent approach*

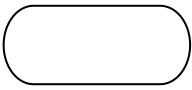
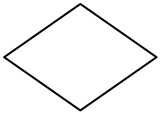
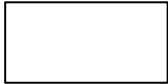
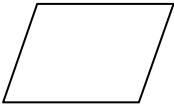
Membuat inferensi yang logis merupakan bagian dari suatu *rational agent*. Hal ini disebabkan satu-satunya cara untuk melakukan aksi secara rasional adalah dengan menalar secara logis. Dengan menalar secara logis, maka bisa didapatkan kesimpulan bahwa aksi yang diberikan akan mencapai tujuan atau tidak. Jika mencapai tujuan, maka agent dapat melakukan aksi berdasarkan kesimpulan tersebut [9].



### 2.6 Flowchart (Bagan Alur)

Flowchart adalah bagan-bagan yang mempunyai arus yang menggambarkan langkah-langkah penyelesaian suatu masalah. Flowchart merupakan cara penyajian dari suatu algoritma. [10]

Berikut adalah simbol-simbol flowchart yang dipakai dalam laporan ini:

Tabel 2.1 Tabel Flowchart

Simbol	Keterangan
	Simbol <i>terminal</i> yaitu menunjukkan permulaan atau akhir suatu program.
	Simbol <i>decision</i> , yaitu menunjukkan suatu kondisi tertentu yang akan menghasilkan dua kemungkinan jawaban : ya / tidak.
	Simbol <i>process</i> , yaitu menyatakan suatu tindakan (proses) yang dilakukan oleh komputer.
	Simbol <i>input/output</i> menyatakan proses input/output tanpa tergantung jenis peralatannya.

	Simbol <i>offline connector</i> , berfungsi menyatakan sambungan dari proses ke proses lainnya dalam halaman yang berbeda.
	Simbol arus / flow, yaitu menyatakan jalannya arus suatu proses.

## 2.7 Storyboard

Storyboard adalah *graphic organizer* yang menyediakan tampilan tingkat tinggi dari proyek yang dikerjakan. [11]

Dalam *Agile software development*, *storyboard* dapat membantu *developer* dengan cepat mengetahui pekerjaan apa yang masih perlu diselesaikan. Selama tim selalu menjaga *storyboard up to date*, siapapun dapat mengetahui pekerjaan apa yang sudah selesai, siapa mengerjakan apa, dan pekerjaan apa yang belum selesai. Ini tidak hanya memberikan pemilik produk dengan transparansi, ini juga membantu tim untuk memvisualisasi urutan dan keterkaitan *user stories*. *Storyboard* bisa berupa fisik ataupun digital.

*Storyboard* berasal dari industri gambar bergerak untuk membantu direktor dan cinematographer untuk memvisualkan adegan film secara berurutan. *Storyboard* seperti itu menyerupai kartun strip. Sementara di *Agile development*, *storyboard* terlihat lebih menyerupai serangkaian kolom yang diisi dengan kertas berwarna dibandingkan menyerupai kartun. Biasanya kolom-kolom tersebut digelar di kertas besar, pada papan tulis, atau papan buletin. Tiap kolom yang merepresentasikan status dan *user stories* digeser ke kolom baru ketika status atau *user stories* berubah.

Pada *Scrum software development*, *storyboard* bisa disebut sebagai papan tugas.

## 2.8 Non Player Character

*Non Player Character*(NPC) adalah semua karakter lain di dalam dunia permainan yang tidak bisa di kendalikan oleh sang pemain, NPC bisa berupa : Monster, pedagang, pemberi tugas (Quest), dan lainnya[12].

## 2.9 Elemen dasar game

Terdapat elemen-elemen dasar dalam pembuatan *game* agar lebih menarik untuk memainkannya. Berikut merupakan elemen dasar *game* [13] :

### 1. *Game Rule*

*Game rule* merupakan aturan perintah, cara menjalankan, fungsi objek dan karakter di dunia permainan dunia *game*. Dunia *game* bisa berupa pulau, dunia khayal, dan tempat-tempat lain yang sejenis yang dipakai sebagai setting tempat dalam permainan *game*.

### 2. *Plot*

*Plot* biasanya berisi informasi tentang hal-hal yang dilakukan oleh player dalam *game* dan secara detail, perintah tentang hal yang harus dicapai dalam *game*.

### 3. *Object*

*Object* merupakan sebuah hal yang penting dan biasanya digunakan pemain untuk memecahkan masalah, adakalanya pemain harus punya keahlian dan pengetahuan untuk bisa memaninkannya.

### 4. *Text*, grafik dan *sound*

*Game* biasanya merupakan kombinasi dari media *text*, grafik maupun *sound*, walaupun tidak harus semuanya ada dalam permainan *game*.

### 5. Animasi

Animasi ini selalu melekat pada dunia *game*, khususnya untuk gerakan karakter-karakter yang ada dalam *game*.

### 6. *User Interface*

Merupakan fitur-fitur yang mengkomunikasikan *user* dengan *game*. *Game* sebaiknya memiliki *user interface* yang menarik dan mudah dimengerti *user* untuk menggunakannya.

## **BAB III. METODOLOGI PENELITIAN**

### **3.1 Metode Pengumpulan Data**

Tahap pengumpulan data yang digunakan dalam pembuatan *game* ini adalah melalui *survey game* yang sejenis dan studi literatur.

#### *1. Survey game sejenis.*

Mencoba game yang sejenis dengan genre game yang akan dibuat untuk memahami cara kerja dari game tersebut.

#### *2. Studi Artikel dan Jurnal.*

Mengumpulkan dan mempelajari sejumlah referensi yang berkaitan dengan algoritma yang akan digunakan.

### **3.2 Metodologi Pengembangan Sistem**

Pengembangan *video game maze “Return To Earth”* dilakukan dengan mengaplikasikan metodologi *Multimedia Development Life Cycle (MDLC)* [13]. Pengembangan multimedia harus melalui tahapan-tahapan yang terancang dengan baik dan runtut agar produk yang dihasilkan memiliki kualitas yangn baik dan tepat digunakan dalam pembelajaran. Tahapan pengembangan dalam *Multimedia Development Life Cycle (MDLC)* ini terdiri dari 6 tahap yaitu :

#### *1. Perencanaan*

Tahapan pada proses ini meliputi pembuatan konsep mengenai *game* yang akan dibuat.

#### *2. Perancangan*

Pada fase *design* (perancangan) dimulai dengan membuat garis besar dari tampilan dan informasi yang akan ditampilkan di layar. Tahap ini biasanya menggunakan *storyboard* untuk menggambarkan deskripsi tiap *scene* lain ke bagan alir (*flowchart*) untuk menggambarkan aliran dari *scene* ke *scene* lain.

### 3. *Material Collecting*

*Material Collecting* (pengumpulan bahan) adalah tahap pengumpulan bahan yang sesuai dengan kebutuhan yang dikerjakan. Bahan-bahan yang dibutuhkan adalah sebagai berikut :

- Audio : *File* audio yang digunakan sebagai musik latar game
- Gambar : Gambar digunakan sebagai objek *player*, *non-player* dan lingkungan *game*.
- *Software* : Kebutuhan perangkat lunak yang dibutuhkan untuk menunjang keperluan pembuatan *game engine*, editor gambar, dan sebagainya

### 4. *Assembly*

Tahap *assembly* (pembuatan) adalah tahap pembuatan semua objek atau bahan multimedia. Pembuatan aplikasi berdasarkan *storyboard*, bagan alir (*flowchart*), dan struktur navigasi yang berasal pada tahap *design*. Proses ini dimulai dengan pemodelan karakter dan lingkungan *game*, pembuatan animasi pada karakter, dan pembuatan *game* serta *source code*.

### 5. *Testing*

Tahap *testing* (pengujian) dilakukan setelah menyelesaikan tahap pembuatan (*assembly*) dengan menjalankan aplikasi / program dan dilihat apakah ada kesalahan atau tidak. Pengujian melibatkan pengguna akhir yaitu para *player*.

### 6. *Distribution*

Tahap ini aplikasi akan disimpan dalam suatu media penyimpanan. Tahap ini juga dapat disebut tahap evaluasi untuk pengembangan produk yang sudah jadi supaya menjadi lebih baik. Hasil evaluasi ini dapat digunakan sebagai masukan untuk tahap *concept* pada produk selanjutnya.

## 3.3 Hasil Survey Game Sejenis

Game yang di *survey* adalah game “*Mummy Maze Deluxe*”. Setelah mencoba game “*Mummy Maze*” diketahui cara kerja dan aturan-aturan yang terdapat di dalam game tersebut. Cara kerja *game* tersebut adalah sebagai berikut :

- ✓ *Game* berupa *turn-based*, dimana karakter *player* dan karakter musuh bergerak secara bergantian.



- ✓ Karakter di dalam game bergerak sesuai *grid*.
- ✓ Karakter *player* hanya bisa bergerak 1 langkah secara *directional*, tidak ada gerakan *diagonal*.
- ✓ Karakter *player* harus menuju pintu keluar dan menghindari karakter musuh untuk menyelesaikan level.
- ✓ Karakter musuh (*mummy*) bergerak mendekati karakter *player*.
- ✓ Karakter musuh dapat bergerak dua langkah.



Gambar 3.1 Tampilan Game Mummy Maze Deluxe



## BAB IV. ANALISIS DAN PERANCANGAN

### 4.1 Analisis

Sebelum dilakukan perancangan dilakukan analisis mengenai gambaran umum sistem yang akan dibangun.

#### 4.1.1 Gambaran Umum Sistem

Sistem *video game* “Return to Earth” menggunakan Algoritma A\* (*A-Star*). Algoritma A\* digunakan untuk mencari jalur antara *NPC* dengan karakter *player*.

#### 4.1.2 Analisis Kebutuhan

Analisis kebutuhan adalah sebuah proses untuk mendapatkan informasi, model, spesifikasi tentang kebutuhan perangkat lunak yang diinginkan klien/pengguna. Kebutuhan *software* yang digunakan dalam membangun *video game* “Return To Earth” adalah :

- Sistem Operasi Windows 7 SP 1+
- *Game Engine* Unity Versi 5.50f
- Inkscape 0.91

Kebutuhan *hardware* yang digunakan dalam membangun *video game* “Return To Earth” adalah :

- Processor 2.5 Ghz
- 8 GB RAM
- *Graphics Card* dengan DX11
- Monitor 1366 x 768
- *Space* HDD 2GB atau lebih
- *Mouse*
- *Keyboard*
- *Speaker*

Kebutuhan minimum *software* dan *hardware* pada *desktop* agar dapat memainkan *video game* “Return To Earth” adalah:

- Monitor minimal 800x600

- *Mouse*
- *Keyboard*
- *Space* HDD 50 MB atau lebih
- *Speaker*

## 4.2 Perancangan

Ada beberapa bagian yang harus dirancang sebelum melakukan implementasi, seperti karakter, *gameplay*, *flowchart* perilaku npc, *flowchart* kondisi menang, serta *storyboard*.

### 4.2.1 Perancangan Karakter

Karakter dalam *game* ini dibedakan menjadi dua, yaitu *player* dan NPC (*non-playable character*). *Player* dalam *game* ini adalah astronot. Berikut adalah desain karakter *player*.



Gambar 4.2 Desain Karakter Game

Karakter NPC adalah alien yang berusaha menangkap astronot yang digerakkan oleh *player*. Berikut adalah desain karakter NPC.



Gambar 4.3 Desain Karakter NPC

### 4.2.2 Perancangan Gameplay

Secara keseluruhan, alur cerita dari *game* ini adalah seorang astronot yang diculik oleh alien dan berusaha untuk kabur dari pesawat alien. Astronot harus berhasil mencapai titik tertentu dengan jumlah langkah kurang dari yang sudah

ditentukan sambil menghindari alien yang berusaha menangkap, apabila melebihi jumlah langkah yang ditentukan, akan datang bala bantuan dari alien yang berarti astronot gagal kabur dan kalah pada level tersebut.

Pada level tertentu, jumlah alien yang mengejar akan meningkat dan meningkatkan tingkat kesulitan dari level tersebut, serta pada level tertentu pula *player* diharuskan mencari kunci yang digunakan untuk mengaktifkan alat teleportasi yang dibutuhkan untuk menyelesaikan level tersebut dan apabila *player* belum mengumpulkan kunci yang dibutuhkan, *player* tidak bisa menyelesaikan level.

*Game* ini terdiri dari 9 level. Pembedanya adalah jumlah alien yang mengejar, jumlah langkah yang disediakan, dan apakah memerlukan kunci untuk mengaktifkan alat teleportasi.

Tabel 4.1 Pembeda Tiap Level

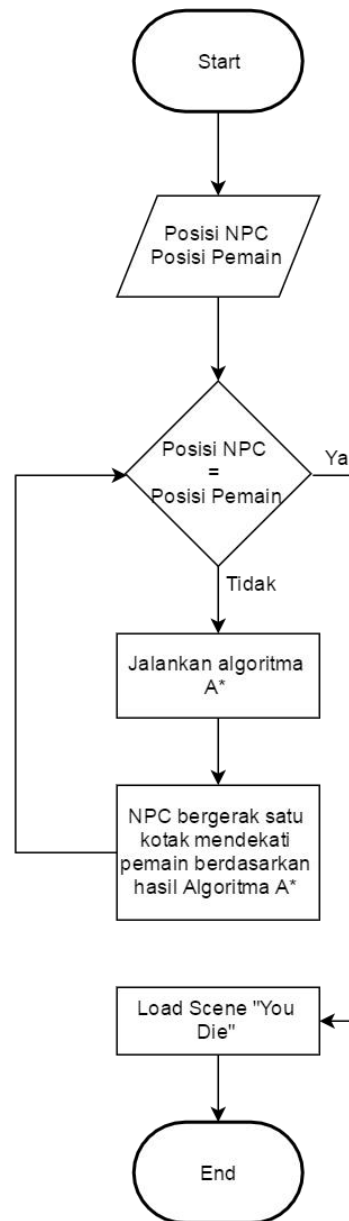
Level	Jumlah Alien	Langkah	Perlu Kunci
Level 1	1	29	Tidak
Level 2	1	26	Tidak
Level 3	1	37	Tidak
Level 4	2	39	Tidak
Level 5	2	23	Tidak
Level 6	2	24	Tidak
Level 7	2	40	Ya
Level 8	2	43	Ya
Level 9	2	48	Ya

Sedangkan ketentuan *scoring* pada *game* ini adalah sebagai berikut :

- 3 Star : Astronot berhasil mengumpulkan seluruh koin dan berhasil menyelesaikan level.
- 2 Star : Astronot berhasil mengumpulkan 2 buah koin dan berhasil menyelesaikan level.
- 1 Star : Astronot berhasil mengumpulkan 1 buah koin dan berhasil menyelesaikan level.
- 0 Star : Astronot berhasil menyelesaikan level.

### 4.2.3 Perancangan Perilaku *Non-Playable Character*

Dalam game “*Return To Earth*” karakter NPC yang mengejar astronot bergerak sesuai dengan jalur terdekat. Jalur ini didapatkan dengan menggunakan algoritma A\*. Berikut adalah *flowchart* perilaku NPC.

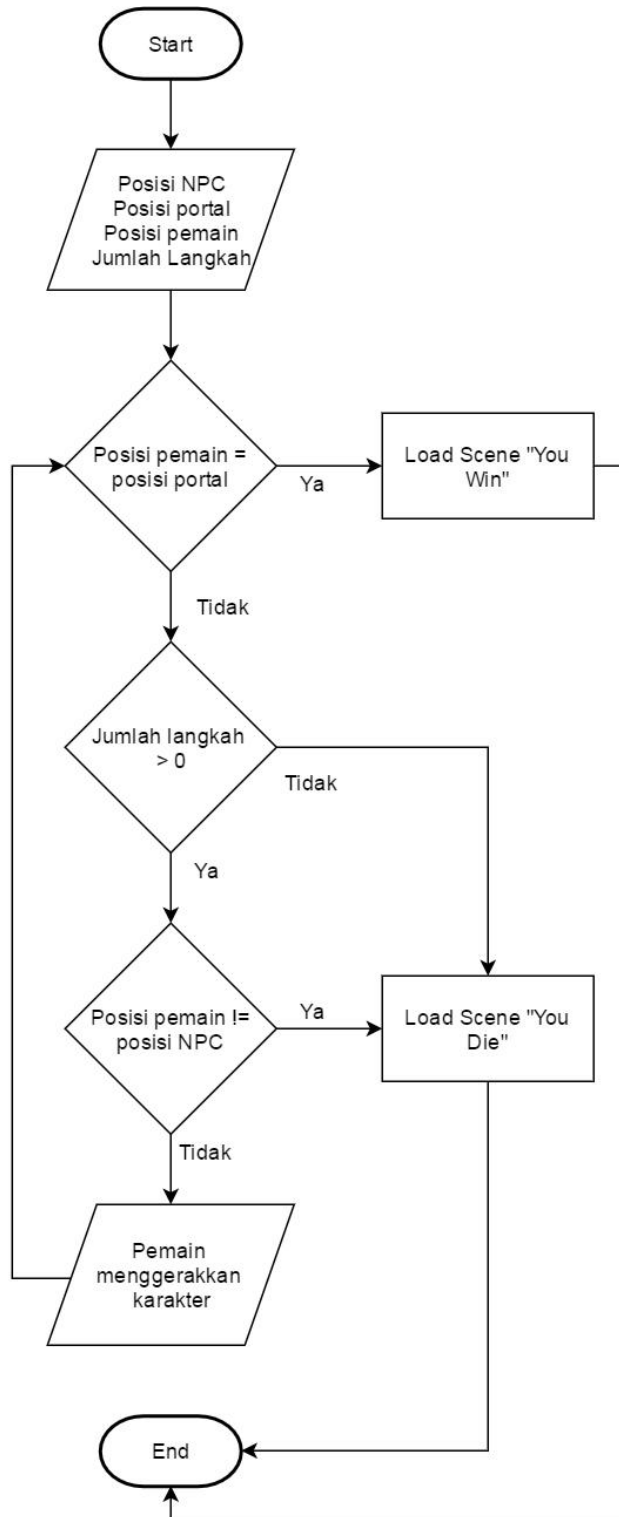


Gambar 4.4 Flowchart Perilaku NPC

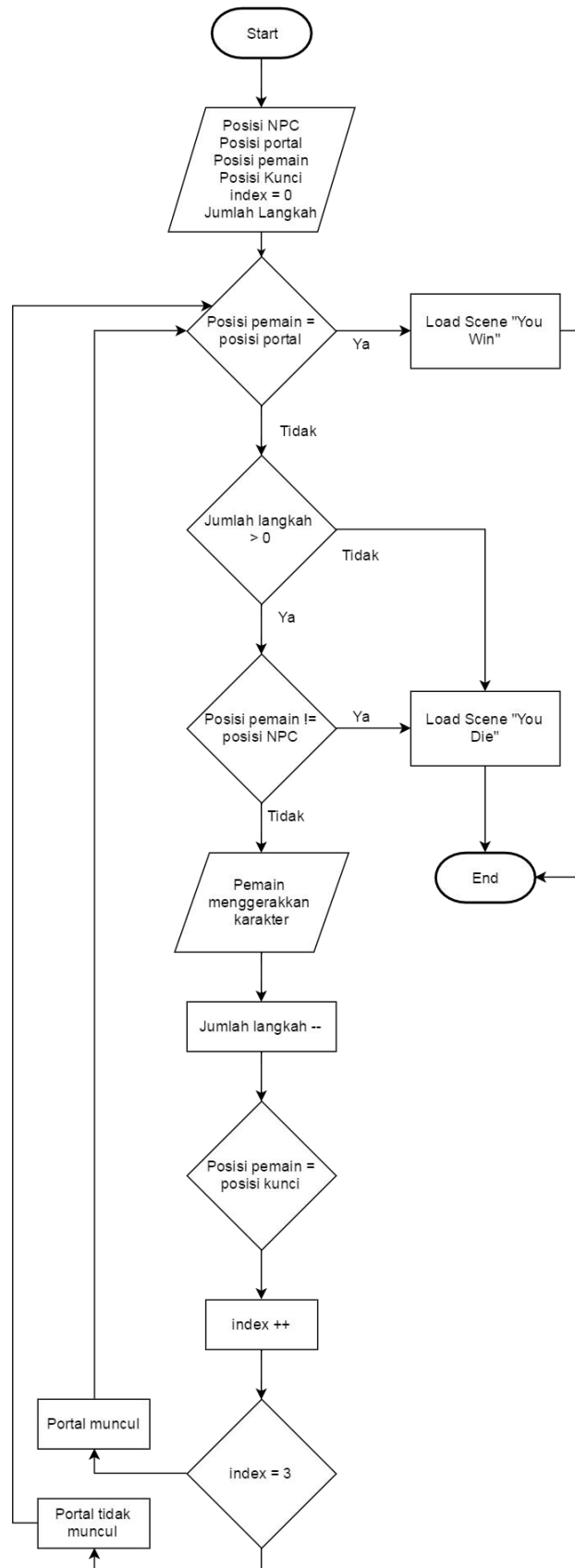
### 4.2.4 Perancangan Kondisi Menang

Dalam game “*Return To Earth*” ada beberapa kondisi yang harus dipenuhi sebelum pemain dapat menyelesaikan level tersebut dan menang. Pada level 1-6 pemain cukup menuju portal untuk menyelesaikan level tersebut, sementara pada

level 7-9 pemain perlu mengumpulkan kunci terlebih dahulu dikarenakan portal tujuan tidak terlihat dari awal permainan. Berikut adalah flowchart kondisi untuk menang.



Gambar 5.4 Flowchart Kondisi Menang Level 1-6

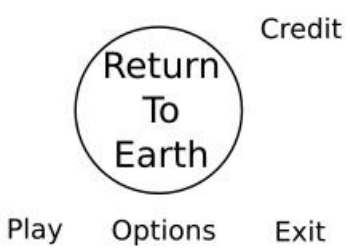
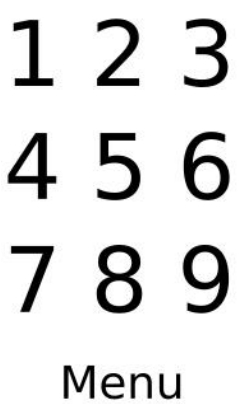



Gambar 4.6 Flowchart Kondisi Menang Level 7-9

#### 4.2.5 Perancangan *Storyboard*

Storyboard adalah visualisasi ide dari game yang akan dibangun sehingga dapat memberikan gambaran dari aplikasi yang akan dihasilkan. Storyboard dapat dikatakan juga visual script yang akan dijadikan outline dari sebuah proyek, ditampilkan shot by shot yang biasa disebut dengan istilah scene. Storyboard sekarang lebih banyak digunakan untuk membuat kerangka pembuatan website dan proyek media interaktif seperti iklan, film pendek, games, ketika dalam tahap perancangan. Berikut adalah perancangan storyboard game “Return To Earth”.

Tabel 4.3 Storyboard

No	Storyboard	Keterangan
1		<p>Ini adalah tampilan awal “<i>Return to Earth</i>”.</p> <p>Pada <i>Main Menu</i> terdapat menu:</p> <ol style="list-style-type: none"> <li>1. Start: Menu untuk memulai permainan</li> <li>2. Option: Menu untuk mengubah pengaturan yang ada di dalam permainan</li> <li>3. Exit: Keluar dari aplikasi</li> <li>4. Credit : Menampilkan panel kredit.</li> </ol>
2		<p>Tampilan setelah player menekan tombol <i>play</i>, muncul tampilan <i>level picker</i> dimana player bisa memilih level yang akan dimainkan. Dan ada tombol menu untuk kembali ke <i>Main Menu</i></p>
3		<p>Tampilan yang akan keluar ketika pemain menekan tombol options.</p> <p>Pemain bisa memilih untuk meniadakan suara pada <i>game</i> ini.</p>

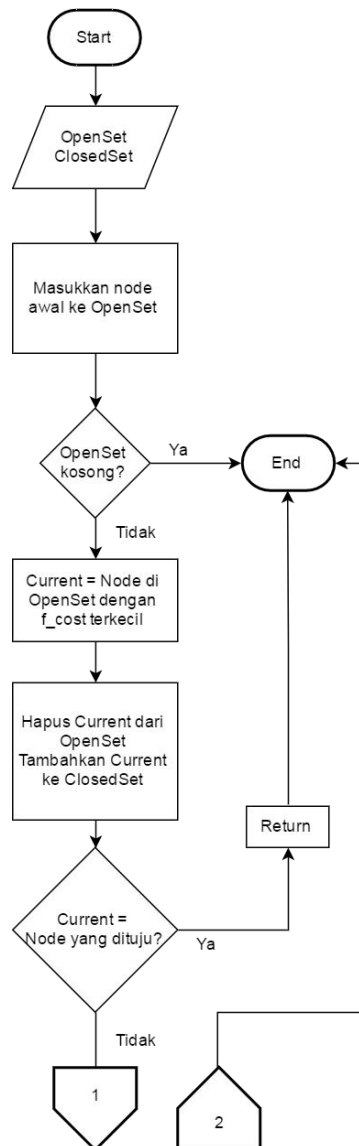
4		<p>Desain level pada <i>game</i> ini bertipe <i>grid</i> berukuran 10x10. Rintangan yang ada pada tiap level dibuat secara manual. Titik biru adalah <i>starting point</i>, titik awal dimana karakter pemain akan keluar dan titik hijau adalah <i>goal</i>, tujuan untuk menyelesaikan level.</p> <p>Sementara titik merah adalah NPC yang harus player hindari.</p>
5		<p>Tampilan yang akan keluar ketika pemain berhasil memenangkan level tersebut.</p> <p>Ada dua tombol, yaitu <i>next level</i> untuk melanjutkan ke level selanjutnya dan <i>main menu</i> untuk kembali ke <i>level picker</i>.</p>
6		<p>Tampilan yang akan keluar ketika pemain kalah pada <i>level</i> tersebut.</p> <p>Ada dua tombol, yaitu <i>retry</i> untuk mengulang level tersebut dan <i>main menu</i> untuk kembali ke <i>level picker</i>.</p>
7		<p>Pada tiap <i>level</i> akan ada koin-koin yang bisa dikumpulkan untuk menambah perolehan bintang yang akan didapat di akhir <i>game</i> setelah pemain berhasil menyelesaikan <i>level</i> tersebut.</p>
8		<p>Pada <i>level</i> tertentu akan muncul kunci, dimana kunci tersebut harus dikumpulkan sebelum titik <i>goal</i> muncul di <i>level</i> tersebut. Apabila kunci belum dikumpulkan, maka <i>goal</i> pada <i>level</i> tersebut tidak akan muncul.</p>



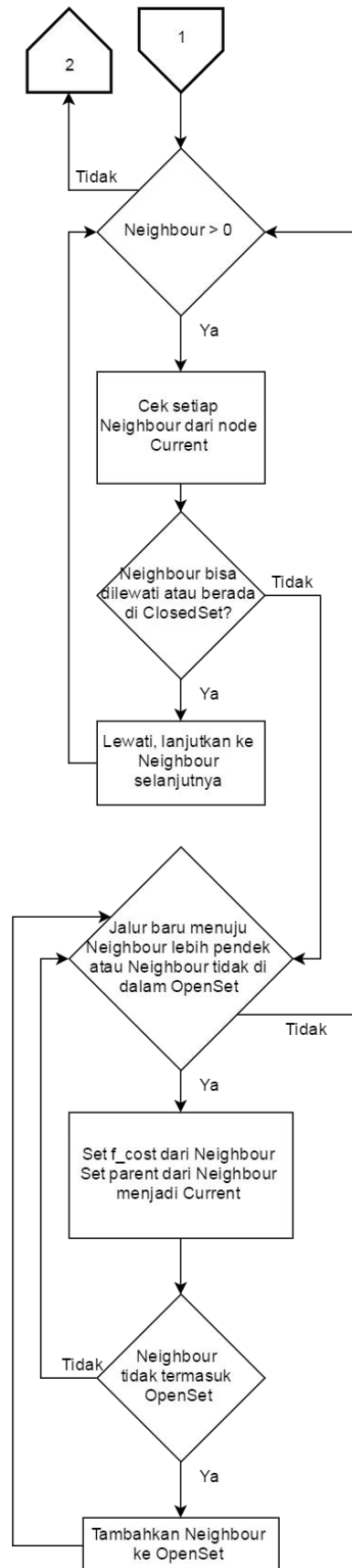
9	<p><b>Credit</b></p> <p>A Game By Ari Mahardika Ahmad Nafis</p> <p>Thanks To Jukedeck</p> <p>Main Menu</p>	<p>Tampilan yang akan keluar ketika pemain menekan tombol <i>credit</i> pada main menu</p>
---	--	--

#### 4.2.6 Perancangan Algoritma

Game “Return To Earth” menggunakan algoritma A\* untuk menentukan bagaimana NPC berjalan. Berikut adalah flowchart dari algoritma A\*



Gambar 4.7 Flowchart A\*



Gambar 4.8 Flowchart A\* (Lanjutan)

## BAB V. IMPLEMENTASI

### 5.1 Implementasi pada Inkscape

Pembuatan *video game maze 2D “Return To Earth”* dimulai dengan pembuatan *game assets* yaitu *sprite* dan gambar lingkungan game. Maksud dari *sprite* adalah gambar-gambar yang mewakili objek dalam suatu *game*. *Software* yang digunakan dalam tahap ini adalah Inkscape 0.91.

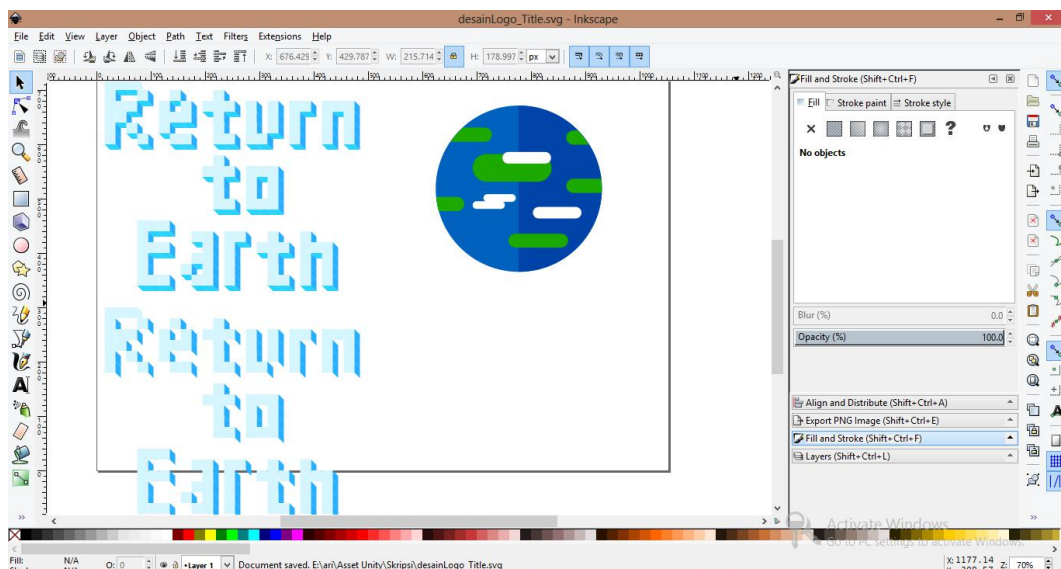


Gambar 5.1 Logo Inkscape

Berikut ini merupakan langkah-langkah pembuatan *asset* yang terdapat pada *game*.

#### 5.1.1 Pembuatan Logo Game

Pembuatan logo *game* dilakukan dengan membuat gambar bumi dan membuat tulisan *Return to Earth* sebagai nama *game*.



Gambar 5.2 Pembuatan Logo Game

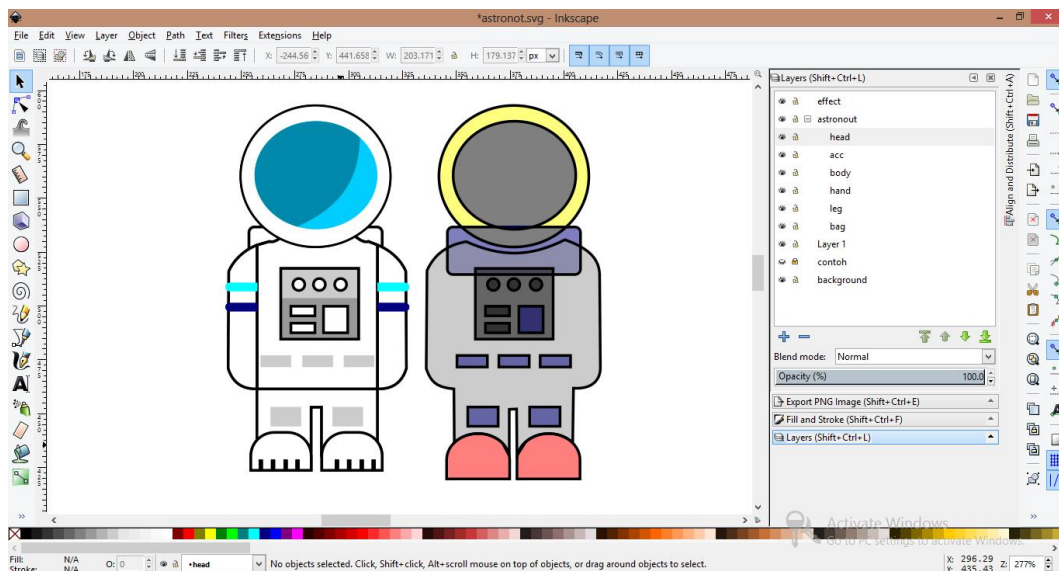
Kemudian teks dan gambar disatukan dan di *export* menjadi satu gambar.



Gambar 5.3 Logo *Game*

### 5.1.2 Pembuatan Karakter *Game*

Pembuatan karakter *game* dimulai dengan mencari beberapa gambar referensi serta membuat sketch pada aplikasi Inkscape yang kemudian diperbaiki menjadi desain karakter yang diinginkan.



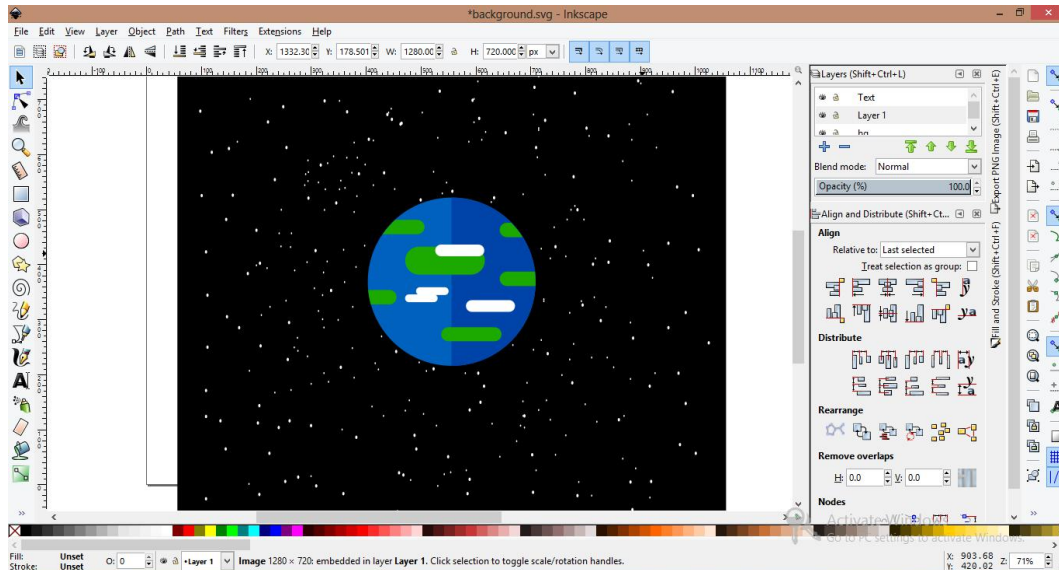
Gambar 5.4 Pembuatan Karakter

### 5.1.3 Pembuatan Lingkungan *Game*

Pembuatan lingkungan *game* ini mencakup pembuatan main menu, tombol-tombol, panel, serta background.

#### 5.1.3.1 *Main Menu*

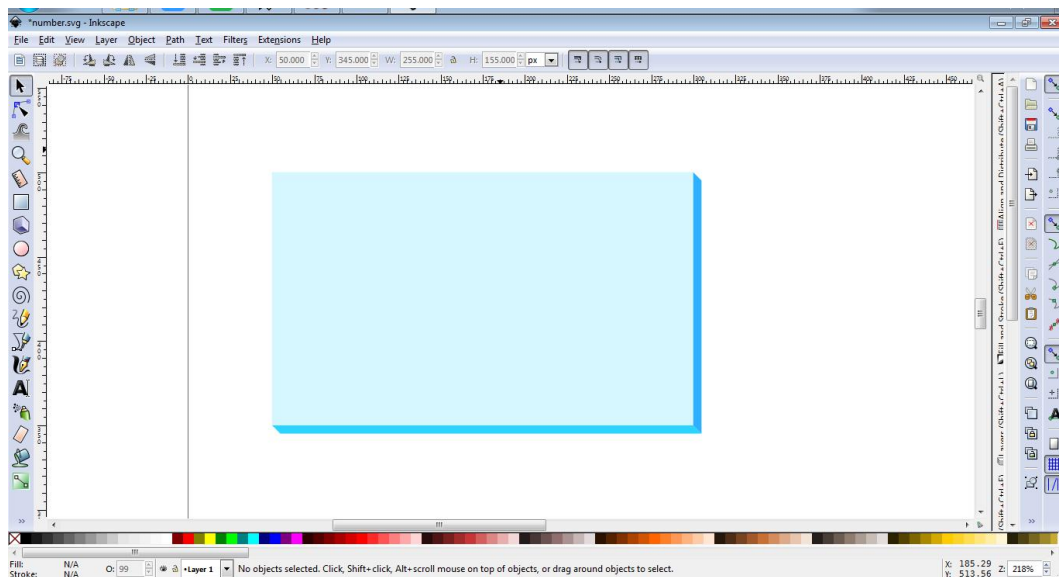
Konsep gambar *main menu* ini adalah gambar Bumi sebagai tempat tujuan dari karakter yang diculik oleh alien.



Gambar 5.5 Desain gambar pada main menu

### 5.1.3.2 *Panel*

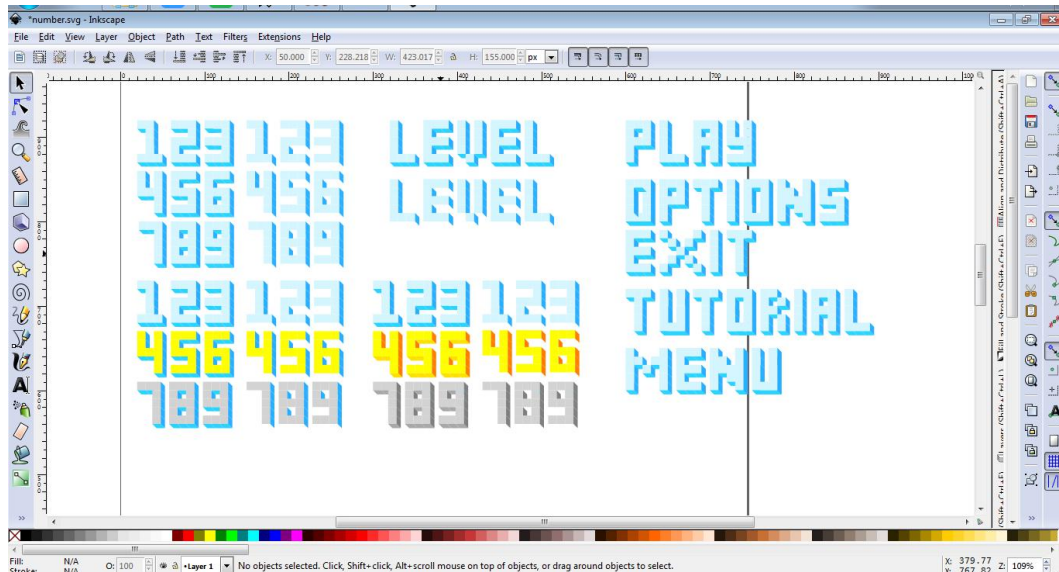
*Panel* merupakan jendela pop-up pada *game*. *Panel* digunakan sebagai tampilan pada saat game di *pause*.



Gambar 5.6 Desain gambar panel

### 5.1.3.3 Tombol-Tombol

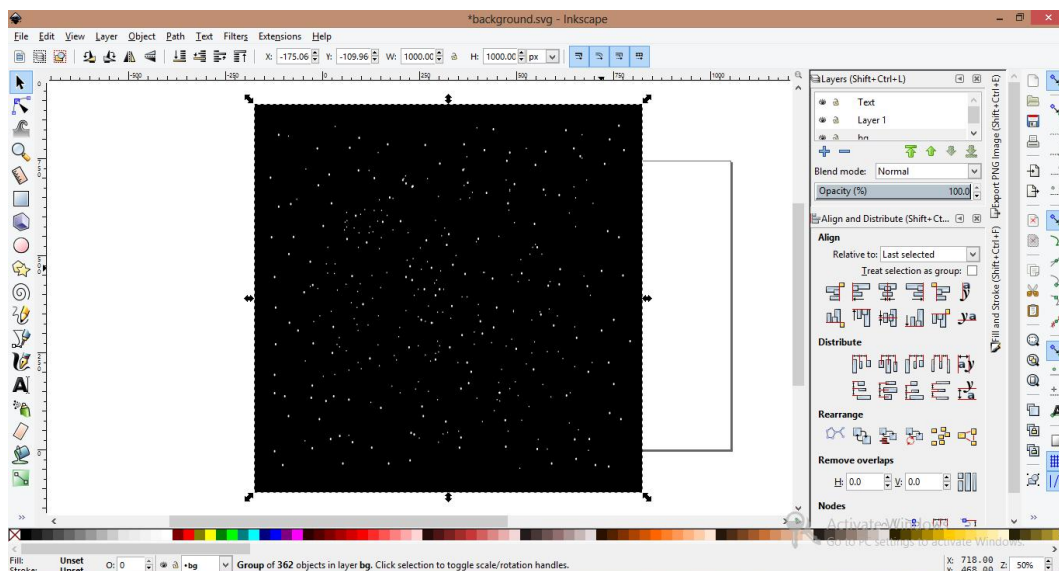
Berikut adalah desain tombol-tombol yang digunakan dalam *game*. Tombol berupa angka digunakan untuk memilih level yang akan dimainkan.



Gambar 5.7 Desain tombol untuk *game*

#### 5.1.3.4 *Background*

*Background* berikut digunakan di seluruh *scene* sebagai dasar dari *game*.

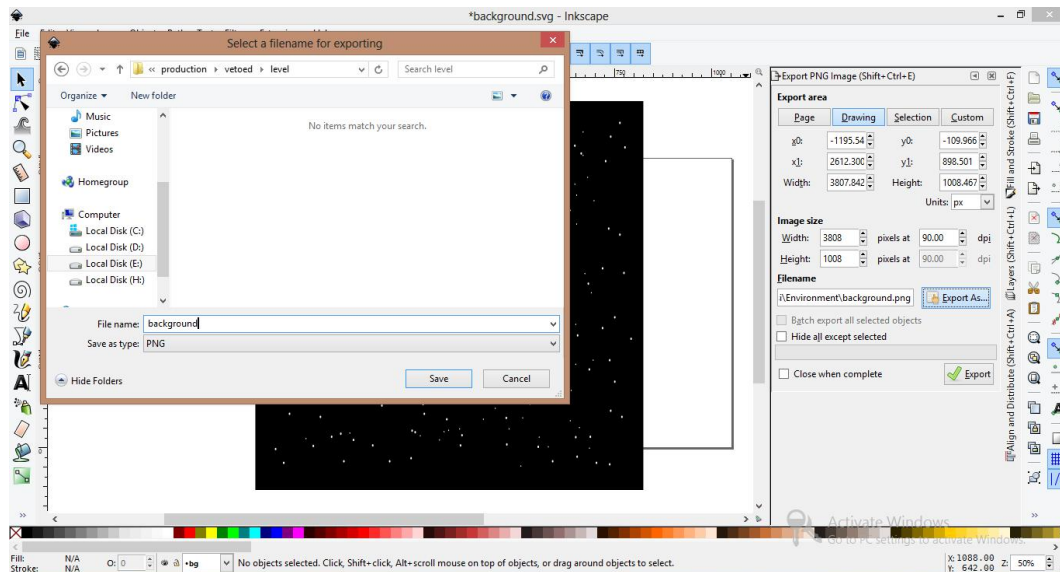


Gambar 5.8 Desain luar angkasa

#### 5.1.4 Mengekspor Gambar Menjadi *Sprite*

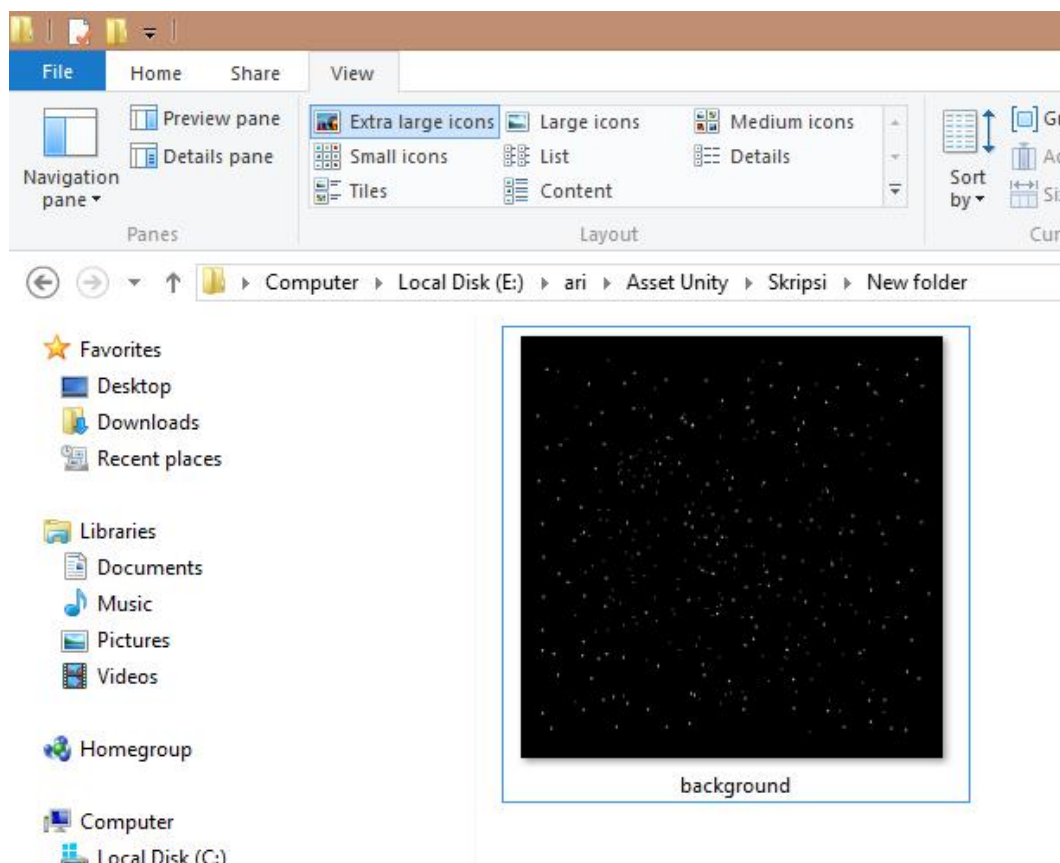
Mengekspor gambar dilakukan agar gambar dapat diakses di Unity. Format file gambar yang digunakan adalah png, karena memiliki fitur *background* yang transparan dibandingkan jpeg.





Gambar 5.9 Mengekspor gambar background.png

Berikut adalah contoh gambar *sprite* yang sudah siap untuk di-import ke *game engine* Unity.



Gambar 5.10 Gambar background.png

## 5.2 Implementasi pada Unity

Aplikasi yang digunakan untuk membangun *game* ini adalah Unity versi 5.5.0f3. Unity merupakan aplikasi yang dapat digunakan untuk membangun *game* 2D maupun 3D. Aplikasi ini cukup mudah digunakan oleh pemula namun juga *powerful* untuk ahli. Unity dapat digunakan siapa saja yang ingin dengan mudah membuat *game* untuk *desktop*, *web*, *mobile*, dan *console*. Bahasa yang digunakan pada unity adalah C# dan *unityscript* (*javascript*).



Gambar 5.11 Logo Unity

### 5.2.1 Main Menu

Pada saat awal *game* dimainkan, *game* langsung menuju *scene main menu*, dimana pemain bisa memilih *Play*, *How To Play*, *Exit*, atau *Credit*.



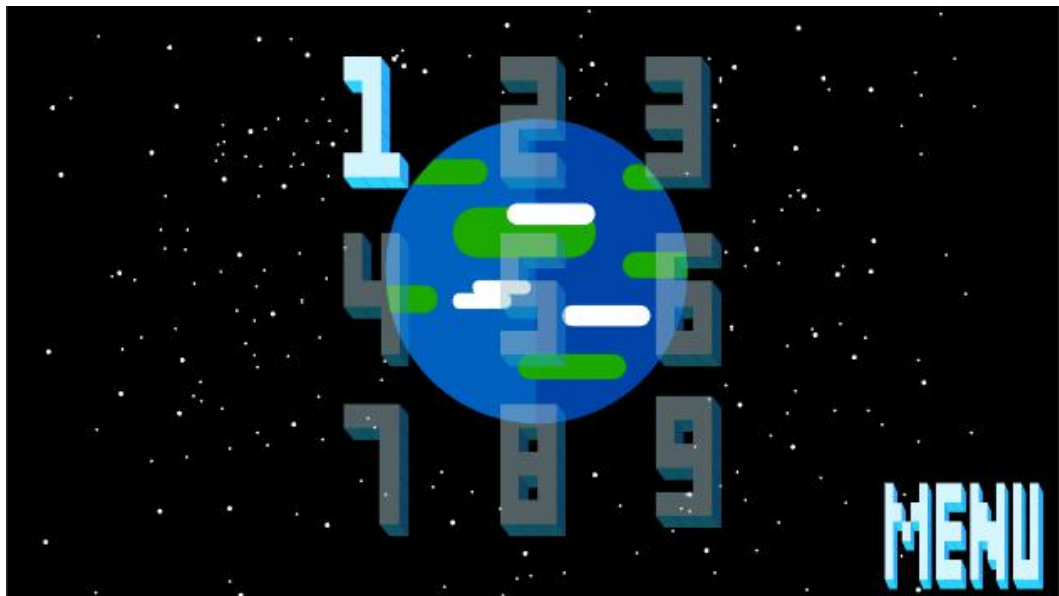
Gambar 5.12 Main Menu



Tombol *play* yang berfungsi untuk berpindah ke *scene level picker*, tombol *how to play* untuk menampilkan bagaimana cara bermain *game* ini, dan tombol *exit* yang berfungsi untuk keluar dari *game*.

### 5.2.2 Level Picker

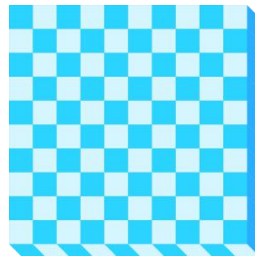
Setelah *player* menekan tombol *play*, akan muncul tampilan dibawah yang digunakan untuk memilih level yang akan dimainkan. Level yang bisa dimainkan akan berwarna cerah, dan level yang belum bisa dimainkan akan berwarna lebih gelap. Player bisa menekan tombol *menu* untuk kembali ke *scene main menu*.



Gambar 5.13 Scene Level Picker

### 5.2.3 Desain Background *Gameboard*

Berikut adalah salah satu background dari *gameboard* dimana player akan menggerakkan karakter yang dimainkan.



Gambar 5.14 Desain Background Gameboard

#### 5.2.4 Penentuan Jumlah Langkah Tiap Level

Untuk menentukan berapa jumlah langkah yang sesuai untuk tiap level, dilakukan percobaan dengan memainkan tiap level, dengan dua cara, menyelesaikan level dengan mengumpulkan koin dan tanpa mengumpulkan koin atau langsung menuju titik tujuan pada level tersebut, dan didapatkan hasil berikut:

Tabel 5.1 Penentuan Jumlah Langkah

Level	Jumlah Langkah Yang Diperlukan	
	Tanpa Koin	Dengan Koin
1	18	26
2	13	23
3	18	34
4	16	36
5	20	20
6	17	21
7	33	37
8	40	40
9	43	45

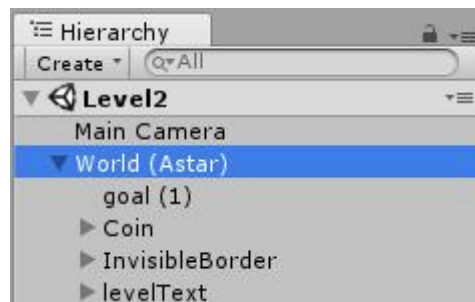
Dari hasil percobaan diatas maka ditentukan bahwa jumlah langkah yang diberikan adalah jumlah langkah terbanyak + 3, untuk mengakomodir tiga kali kesalahan.

Tabel 5.2 Langkah Yang Disediakan Untuk Tiap Level

Level	Langkah
Level 1	29
Level 2	26
Level 3	37
Level 4	39
Level 5	23
Level 6	24
Level 7	40
Level 8	43
Level 9	48

### 5.2.5 Implementasi Algoritma A\*

Implementasi Algoritma A\* diletakkan pada *GameObject* kosong yang digunakan untuk menampung *script* yang digunakan untuk mengimplementasikan algoritma A\*.



Gambar 5.15 Hierarchy Game “Return To Earth”



Gambar 5.16 *Inspector World(Astar)*

Dari gambar diatas diketahui bahwa *GameObject World(Astar)* memiliki 3 buah *script* dimana 2 *script* tersebut merupakan bagian dari algoritma A\*, yaitu *Grid* dan *Pathfinding*. *Script Grid* digunakan untuk menentukan posisi *player*, NPC, serta layer apa yang dianggap tidak bisa dilewati. *Grid* juga menentukan ukuran papan yang digunakan dalam level. *Node radius* digunakan untuk mendefinisikan seberapa besar cakupan tiap *node*.

*Script Pathfinding* digunakan untuk memperhitungkan jalur terdekat dari karakter NPC menuju karakter *player*.

```
public class Pathfinding : MonoBehaviour {

    public Transform seeker, target;
    public GameObject player;

    Grid grid;

    void Awake() {
        grid = GetComponent<Grid> ();
    }

    void Update() {
        if (seeker) {
            FindPath (seeker.position, target.position);
        }
    }
}
```

```

    }
}

void FindPath(Vector2 startPos, Vector2 targetPos){ . . .
}

public void RetracePath(Node startNode, Node endNode){ . . .
}

int GetDistance(Node nodeA, Node nodeB){ . . .
}

public Vector2 GetCoordinate(Vector2 startPos, Vector2
targetPos){ . . .
}
}

```

Gambar 5.17 Fungsi pada *script Pathfinding*

Sementara script *MoveNPC* digunakan untuk menggerakkan karakter NPC sesuai dengan jalur yang didapatkan dari script *Pathfinding* serta memberikan *delay* untuk *input* dari *player*.

```

public class MoveNPC : MonoBehaviour {

    Pathfinding path;

    public Transform player, npc;
    public GameObject playerObject;

    public float timeBetweenKeyPress = 0.5f;
    private float delayTime;

    void Awake(){
        path = GetComponent<Pathfinding> ();
    }

    void Update () {
        if(Time.time >= delayTime && (
            Input.GetKeyDown (KeyCode.W) || Input.GetKeyDown
            (KeyCode.A) || Input.GetKeyDown (KeyCode.S) || Input.GetKeyDown
            (KeyCode.D) ||
                Input.GetKeyDown
            (KeyCode.UpArrow) || Input.GetKeyDown
            (KeyCode.LeftArrow) || Input.GetKeyDown
            (KeyCode.DownArrow) || Input.GetKeyDown (KeyCode.RightArrow))
            && npc.position != player.position)) {

            Vector2 newPos = path.GetCoordinate (player.position,
            npc.position);
            npc.position = newPos;
            delayTime = Time.time + timeBetweenKeyPress;
        }
    }
}

```

Gambar 5.18 Fungsi pada *script MoveNPC*

## BAB VI. PENGUJIAN DAN PEMBAHASAN

### 6.1 Pengujian

#### 6.1.1 Pengujian *Alpha*

Pengujian dilakukan terhadap aplikasi untuk memastikan bahwa aplikasi dapat berjalan dengan benar sesuai dengan kebutuhan dan tujuan yang diharapkan. Pengujian alpha berfokus pada persyaratan fungsional perangkat lunak.

##### 6.1.1.1 Kasus dan Hasil Pengujian *Alpha*

Pengujian ini dilakukan secara *black box* dengan hanya memperhatikan masukan ke dalam sistem dan keluaran dari masukan tersebut. Berikut adalah hasil pengujian dari game *Return To Earth*.

Tabel 6.1 Tabel Kasus Uji

No	Kasus yang Diuji	Hasil yang Diharapkan	Hasil Pengujian
<b><i>Scene Logo Game</i></b>			
1	Tombol Start	Menampilkan Menu Utama	Berjalan Baik
<b><i>Scene Main Menu</i></b>			
2	Tombol Play	Menampilkan <i>Scene Level Picker</i>	Berjalan Baik
3	Tombol Setting	Menampilkan Panel Options	Berjalan Baik
4	Tombol Exit	Menutup Aplikasi	Berjalan Baik
5	Tombol Credit	Menampilkan <i>Scene Credit</i>	Berjalan Baik
<b><i>Scene Level Picker</i></b>			
6	Tombol Level 1	Menampilkan level 1	Berjalan Baik
7	Tombol Level 2	Tidak aktif jika level sebelumnya belum diselesaikan	Berjalan Baik
8	Tombol Level 2	Aktif jika level sebelumnya sudah diselesaikan	Berjalan Baik
9	Tombol Level 2	Menampilkan Level 2	Berjalan Baik
10	Tombol Level 3	Tidak aktif jika level sebelumnya belum diselesaikan	Berjalan Baik
11	Tombol Level 3	Aktif jika level sebelumnya sudah diselesaikan	Berjalan Baik

12	Tombol Level 3	Menampilkan Level 3	Berjalan Baik
13	Tombol Level 4	Tidak aktif jika level sebelumnya belum diselesaikan	Berjalan Baik
14	Tombol Level 4	Aktif jika level sebelumnya sudah diselesaikan	Berjalan Baik
15	Tombol Level 4	Menampilkan Level 4	Berjalan Baik
16	Tombol Level 5	Tidak aktif jika level sebelumnya belum diselesaikan	Berjalan Baik
17	Tombol Level 5	Aktif jika level sebelumnya sudah diselesaikan	Berjalan Baik
18	Tombol Level 5	Menampilkan Level 5	Berjalan Baik
19	Tombol Level 6	Tidak aktif jika level sebelumnya belum diselesaikan	Berjalan Baik
20	Tombol Level 6	Aktif jika level sebelumnya sudah diselesaikan	Berjalan Baik
21	Tombol Level 6	Menampilkan Level 6	Berjalan Baik
22	Tombol Level 7	Tidak aktif jika level sebelumnya belum diselesaikan	Berjalan Baik
23	Tombol Level 7	Aktif jika level sebelumnya sudah diselesaikan	Berjalan Baik
24	Tombol Level 7	Menampilkan Level 7	Berjalan Baik
25	Tombol Level 8	Tidak aktif jika level sebelumnya belum diselesaikan	Berjalan Baik
26	Tombol Level 8	Aktif jika level sebelumnya sudah diselesaikan	Berjalan Baik
27	Tombol Level 8	Menampilkan Level 8	Berjalan Baik
28	Tombol Level 9	Tidak aktif jika level sebelumnya belum diselesaikan	Berjalan Baik
29	Tombol Level 9	Aktif jika level sebelumnya sudah diselesaikan	Berjalan Baik
30	Tombol Level 9	Menampilkan Level 9	Berjalan Baik
<b>Scene Level</b>			
31	<i>Turn Remaining</i>	Berkurang setiap kali pemain menekan tombol untuk bergerak	Berjalan Baik
32	Perolehan Koin	Koin yang diambil hilang dan indikator koin menyala.	Berjalan Baik

33	Player menyentuh NPC	Menampilkan <i>scene</i> “ <i>You Die</i> ”	Berjalan Baik
34	Player menyentuh portal	Menampilkan <i>scene</i> “ <i>You Win</i> ”	Berjalan Baik
<b>Scene You Win</b>			
35	Tombol Next Level	Menampilkan level selanjutnya	Berjalan Baik
36	Tombol Return to Menu	Menampilkan <i>scene Level Picker</i>	Berjalan Baik
<b>Scene You Die</b>			
37	Tombol Retry	Menampilkan level yang dimainkan sebelumnya	Berjalan Baik
38	Tombol Return to Menu	Menampilkan <i>scene Level Picker</i>	Berjalan Baik
<b>Scene Credit</b>			
39	Tombol Menu	Menampilkan <i>scene main menu</i>	Berjalan Baik

#### 6.1.1.2 Kesimpulan Pengujian *Alpha*

Berdasarkan hasil pengujian diatas maka dapat ditarik kesimpulan bahwa sistem *game Return To Earth* sudah berjalan seperti yang diharapkan dan secara fungsional sudah dapat menghasilkan keluaran yang diharapkan.

#### 6.1.2 Pengujian Beta

Pengujian beta dilakukan dengan melakukan pengujian melibatkan para pengguna aplikasi *game* secara langsung dengan menggunakan kuesioner mengenai algoritma A\* dan map yang dibuat dan melihat respon *user* terhadap pergerakan NPC.

##### 6.1.2.1 Kuesioner Pengujian Beta

Berikut adalah kuesioner yang diisi oleh *user* setelah mencoba *video game* “*Return to Earth*”.



Tabel 6.2 Kuesioner Pengujian Beta

No	Pertanyaan	Nilai				
		5	4	3	2	1
Game						
1	Tahapan antar level					
2	Kesulitan tiap level					
3	Pengambilan jalur NPC					
4	Desain Visual					
User						
1	Kemampuan Memainkan Game					

Berikut adalah keterangan kolom nilai pada kuesioner

Tabel 6.3 Keterangan nilai kuesioner

5	4	3	2	1
Sangat Baik	Baik	Cukup	Kurang	Sangat Kurang

#### 6.1.2.2 Hasil Pengujian Beta

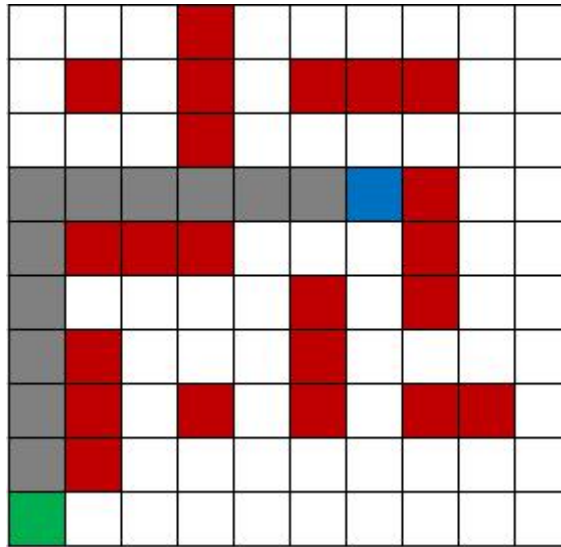
Berikut adalah hasil kuesioner pengujian beta.

Tabel 6.4 Hasil Kuesioner

No	Nama	<i>Game</i>				<i>User</i>
		#1	#2	#3	#4	#1
1	Meilinda Ika Zuliyati	5	5	5	3	3
2	Achmad Prayogi	5	5	5	4	3
3	Danurdara Pradnya Baswara	5	4	3	3	4
4	Anggi Kurniawan	4	4	5	4	3
5	Dhike Almira	4	4	5	2	3
6	Dovie Yudhawiratama	5	4	5	5	3
7	Pipik Lestari	4	4	5	3	3
8	M. Akbar H.	4	4	2	3	4
9	Afwika Chori Q	5	5	5	4	4
10	Ersa Rahmawati	4	5	4	4	4
<b>Rata-rata</b>		<b>4.5</b>	<b>4.4</b>	<b>4.4</b>	<b>3.5</b>	<b>3.4</b>

### 6.1.3 Pengujian Algoritma A\*

Untuk menguji apakah algoritma A\* ini berjalan sesuai dengan hasil yang diharapkan, pengujian algoritma A\* dilakukan dengan melakukan pengecekan secara manual dengan menggunakan perhitungan untuk menentukan apakah jalur yang diambil oleh NPC sudah optimal atau tidak pada setiap level pada *game*.



Gambar 6.1 Pengujian Algoritma A\*

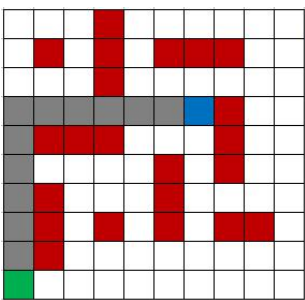
Pada gambar diatas garis abu-abu adalah jalur yang akan diambil oleh NPC, titik biru adalah NPC, titik merah adalah penghalang, dan titik hijau adalah lokasi karakter *player* yang merupakan tujuan dari NPC. Algoritma yang diimplementasikan pada NPC sudah bekerja dengan baik dengan membuat jalur yang menghindari penghalang dan menuju pada karakter *player*.

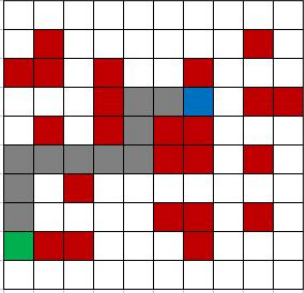
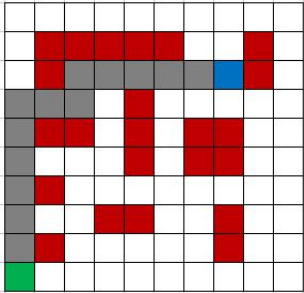
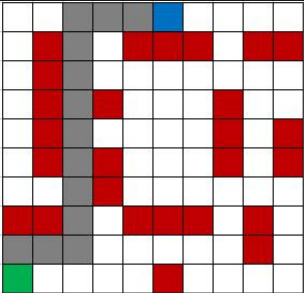
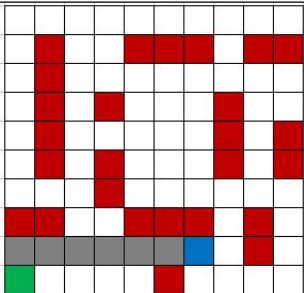
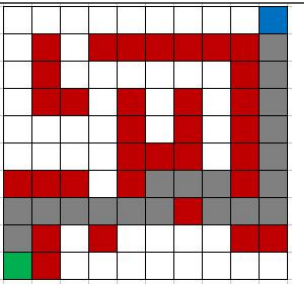
#7 150 70 80	#6 150 60 90	#5 150 50 100		#3 150 30 120	#2 150 20 130	#1 150 10 140			
#6 130 60 70	#5 130 50 80	#4 130 40 90	#3 130 30 100	#2 130 20 110	#1 130 10 120				
#7 130 70 60				#3 130 30 100	#2 130 20 110	#1 130 10 120			
#8 130 80 50	#7 130 70 60	#6 130 60 70	#5 130 50 80	#4 130 40 90	#2 130 20 110				
#9 130 90 40		#7 130 70 60	#6 130 60 70	#5 130 50 80	#3 130 30 100	#4 150 40 110			
#10 130 100 30		#8 130 80 50		#6 130 60 70	#4 130 40 90				
#11 130 110 20		#9 130 90 40	#8 130 80 50	#7 130 70 60	#6 130 60 70	#5 130 50 80	#6 150 60 90		
#12 120 120 0	#11 130 110 20	#10 130 100 30	#9 130 90 40	#8 130 80 50	#7 130 70 60	#6 130 60 70	#7 150 70 80		

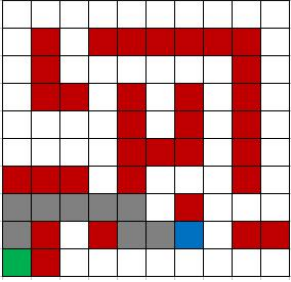
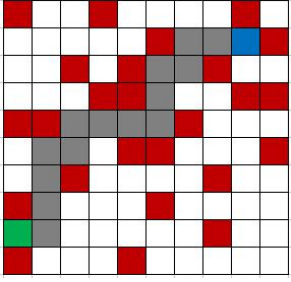
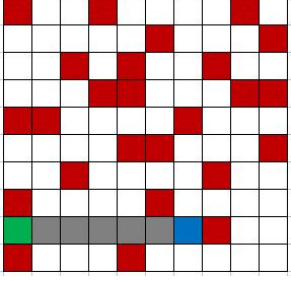
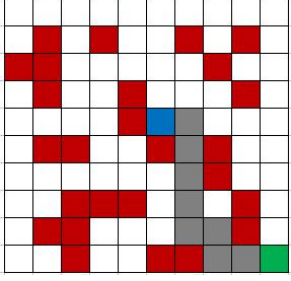
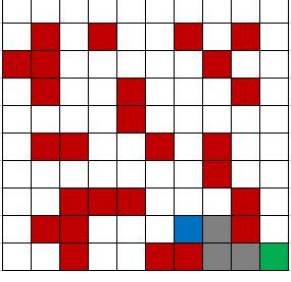
Gambar 6.2 Pengujian Algoritma Menggunakan Perhitungan Manual

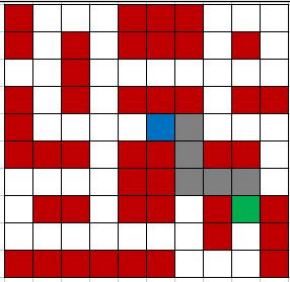
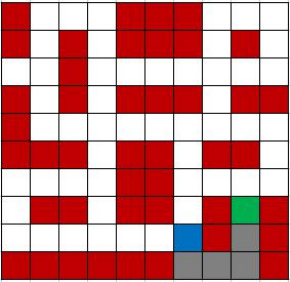
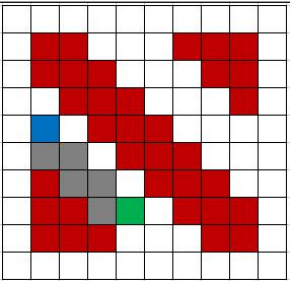
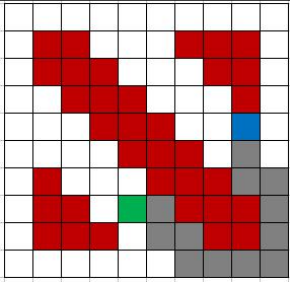
Pada gambar diatas, dilakukan pengujian algoritma A\* menggunakan perhitungan untuk melihat apakah jalur yang diambil sudah optimal, kotak biru merupakan NPC, kotak hijau tua merupakan karakter *player*, kotak hijau muda merupakan jalur yang bisa digunakan oleh algoritma A\* sementara kotak oranye merupakan jalur yang tidak diambil oleh algoritma A\*.

Tabel 6.4 Pengujian Algoritma A\*

Level/ NPC	Map	Jumlah Langkah	Hasil Perhitungan Manual	Hasil
1 / NPC 1		12	12	Optimal

2 / NPC 1		11	11	Optimal
3 / NPC 1		14	14	Optimal
4 / NPC 1		14	14	Optimal
4 / NPC 2		7	7	Optimal
5 / NPC 1		20	20	Optimal

5 / NPC 2		9	9	Optimal
6 / NPC 1		15	15	Optimal
6 / NPC 2		6	6	Optimal
7 / NPC 1		9	9	Optimal
7 / NPC 2		4	4	Optimal

8 / NPC 1		6	6	Optimal
8 / NPC 2		6	6	Optimal
9/ NPC 1		6	6	Optimal
9/ NPC 2		13	13	Optimal

## 6.2 Pembahasan

Hasil yang didapat dari pengujian algoritma A\* pada tiap level didalam game menggunakan perhitungan secara manual adalah jalur yang didapatkan oleh A\* sudah optimal dan merupakan jalur terpendek untuk menuju tujuan.

Hasil dari pengujian beta adalah penilaian terhadap tahapan antar level menurut *user* adalah 4.5 dari nilai maksimal 5, sedangkan tingkat kesulitan tiap level adalah 4.4 dari nilai maksimal 5. Sementara kemampuan *user* dalam bermain *game* ini adalah 3.4 dari nilai maksimal 5.

### 6.2.1 Tingkat Keberhasilan Algoritma A\*

Untuk mengukur tingkat keberhasilan dari implementasi algoritma A\* pada NPC dilakukan penghitungan berapa banyak *user* yang memberi nilai diatas tiga pada pernyataan di bagian *game* pada tabel 6, yaitu pengambilan jalur NPC. Dari hasil kuesioner didapatkan nilai rata-rata 4.4 yang berarti algoritma yang diimplementasikan sudah berhasil dan bekerja dengan baik.

## **BAB VII. KESIMPULAN**

### **7.1 Kesimpulan**

Kesimpulan yang didapatkan dari pengujian adalah A\* berhasil diimplementasikan dengan baik dan berhasil menemukan jalur terpendek untuk setiap level. Serta hasil yang didapatkan dari kuesioner menunjukkan bahwa pengambilan jalur NPC sangat baik berdasarkan nilai yang diberikan oleh user. Dengan demikian algoritma A\* ini dapat digunakan untuk mengambil jalur terdekat dari NPC menuju player.

### **7.2 Saran**

*Video game "Return to Earth"* ini masih bisa dikembangkan lebih jauh dari banyak sisi, *video game* ini dapat dikembangkan dari sisi :

- ✓ Pengambilan jalur NPC bisa dikembangkan menggunakan metode pencarian jalur yang lain.
- ✓ Sisi tampilan atau desain visual, dengan memberikan variasi dari NPC serta penghalang.
- ✓ Dari sisi *gameplay* dengan menambah jumlah level yang tersedia, serta mengubah platform dari *PC* ke *mobile*.




## DAFTAR PUSTAKA

- [1] Harianja, Firman. 2013. Penerapan Algoritma A\* Pada Permasalahan Optimalisasi Pencarian Solusi Dynamic Water JUG. Medan: STMIK Budidarma Medan
- [2] Irsyad, Muhammad and Rasilla, Endang “Aplikasi Pencarian Lokasi Gedung dan Ruangan Universitas Islam Negeri Sultan Syarif Kasim Riau pada Platform Android Menggunakan Algoritma A-Star (A\*)”
- [3] Purwati, Mutia and Firnawati, Okti and Willy, Willy “PENERAPAN ALGORITME A\* (A STAR) DALAM OPTIMASI PENENTUAN HALTE TRANSMUSI DI PALEMBANG BERBASIS ANDROID.” STMIK GI MDP.
- [4] Suyanto, 2014, Artificial Intelligence Searching-Reasoning-Planning-Learning : 2.3.2.6 A\* (A Bintang), Revisi Kedua, Informatika Bandung :Bandung, 2014, pp33.
- [5] Raharjo, Budi “.NET Framework dan C#” in Mudah Belajar C# (Pemrograman C# dan Visual C#). Bandung:Informatika Bandung, 2015, Bab 1. pp.3
- [6] Zamojc, Ian. (2012, May.17) Introduction to Unity3D [Online]. Available : <https://code.tutsplus.com/tutorials/introduction-to-unity3d--mobile-10752>
- [7] Inkscape Org, About Inkscape[Online]. Available <https://inkscape.org/en/about/>
- [8] Kusumadewi, Sri, Artificial Intelligence (Teknik dan Aplikasinya), Graha Ilmu, Yogyakarta, 2003.
- [9] Suyanto, 2014, Artificial Intelligence Searching-Reasoning-Planning-Learning : 1.2 Definisi AI, Revisi Kedua, Informatika Bandung :Bandung, 2014, pp3-4.
- [10] Sulistyorini, Tri, Algoritma dan Pemrograman 1 [Online]. Available: [http://tri\\_s.staff.gunadarma.ac.id/Downloads/files/15392/2+definisi+dan+simbol+Flowchart.pdf](http://tri_s.staff.gunadarma.ac.id/Downloads/files/15392/2+definisi+dan+simbol+Flowchart.pdf).
- [11] Rouse, Margaret, (2013, August) Storyboard [Online]. Available: <http://whatis.techtarget.com/definition/storyboard>


- [12] Pangajaw, Frank Albert, 2008 “RPG Studio”.Jakarta:Elex Media Komputindo
- [13] Paliguna,I Made Adi.” RANCANG BANGUN *GAME TOWER DEFENSE* “THE LEGEND OF KEBO IWA” BERBASIS ANDROID MENGGUNAKAN ALGORITMA A STAR” Universitas Udayana Bukit Jimbaran Bali
- [14] Rasel, 2014, *Multimedia System Development Life Cycle (MSDLC)*. [Online]. Available: <http://bankofinfo.com/multimedia-system-development-life-cycle-msdlc/>
- [10 Desember 2015]

## LAMPIRAN-LAMPIRAN

### Lampiran 1 Lembar Bimbingan Dosen Pembimbing 1










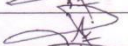



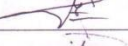





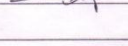

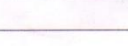
**KEMENTRIAN RISET, TEKNOLOGI DAN PENDIDIKAN TINGGI**  
**POLITEKNIK NEGERI MALANG**  
**JURUSAN TEKNOLOGI INFORMASI**  
**PROGRAM STUDI TEKNIK INFORMATIKA**  
 JL. Soekarno Hatta PO Box 04 Malang Telp. (0341) 404424 pes. 1122

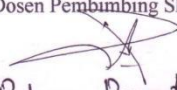


**NO SKRIPSI:190**


**LEMBAR BIMBINGAN SKRIPSI 2016/2017**

**JUDUL** : RANCANG BANGUN GAME MAZE 2D "RETURN TO EARTH"  
**Nama** : ARI MAHARDIKA AHMAD NAFIS **NIM** : 1341180068


No.	Tanggal	Materi Bimbingan	Tanda Tangan	
			Mahasiswa	Dosen
1.	8/03/17	Engine Game		
2.	22/03/17	Desain Gambar Game		
3.	29/03/17	Isometric		
4.	06/04/17	Isometric		
5.	06/04/17	Desain Map		
6.	13/04/17	Laporan		
7.	19/04/17	Flauchart		
8.	17/05/17	Revisi Laporan		
9.	19/05/17	Revisi Laporan		
10.	02/06/17	Revisi Laporan		
11.				
12.				
13.				
14.				
15.				
16.				
17.				
18.				
19.				

Malang, .....  
 Dosen Pembimbing Skripsi,  
  
Ridwan Rismanto, S.ST., M.Kom  
 NIP. 1986 03 18 2012 21001

Lampiran 2 Lembar Bimbingan Dosen Pembimbing 1



**KEMENTERIAN RISET, TEKNOLOGI DAN PENDIDIKAN TINGGI**  
**POLITEKNIK NEGERI MALANG**  
**JURUSAN TEKNOLOGI INFORMASI**  
**PROGRAM STUDI TEKNIK INFORMATIKA**  
 JL. Soekarno Hatta PO Box 04 Malang Telp. (0341) 404424 pes. 1122


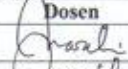

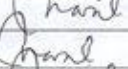

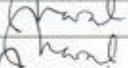

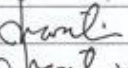

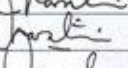

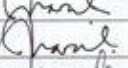

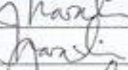

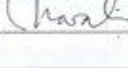

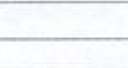



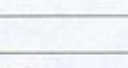



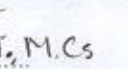


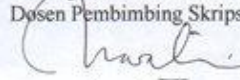
NO SKRIPSI:190

**LEMBAR BIMBINGAN SKRIPSI 2016/2017**

**JUDUL** : RANCANG BANGUN GAME MAZE 2D "RETURN TO EARTH"  
**Nama** : ARI MAHARDIKA AHMAD NAFIS

**NIM** : 1341180068

No.	Tanggal	Materi Bimbingan	Tanda Tangan	
			Mahasiswa	Dosen
1.	7/3/2017	Engine		
2.	05/09/17	Desain (Gambar)		
3.	05/09/17	Desain (Gambar)		
4.	05/09/17	Desain (Warna)		
5.	05/09/17	Desain (Warna)		
6.	02/05/17	Laporan Bab 4		
7.	02/05/17	Laporan Bab 4		
8.	05/05/17	Laporan Bab 6		
9.	09/05/17	Tupuan		
10.	09/05/17	Kesimpulan		
11.	16/05/17	Acc Laporan		
12.	16/05/17	Acc Laporan		
13.	16/05/17	acc maju ujian		
14.				
15.				
16.				
17.				
18.				
19.				

Malang.....  
 Dosen Pembimbing Skripsi,  
  
 Duah Ayu Irawati ST, M.Cs  
 NIP. 198407082008122001

## Lampiran 3 Listing Program

### *Script 1. Pathfinding.cs*

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Pathfinding : MonoBehaviour {

    public Transform seeker, target;
    public GameObject player;

    Grid grid;

    void Awake() {
        grid = GetComponent<Grid> ();
    }

    void Update() {
        if (seeker) {
            FindPath (seeker.position, target.position);
        }
    }

    void FindPath(Vector2 startPos, Vector2 targetPos){

        Node startNode = grid.GetNodeFromWorldPoint (startPos);
        Node targetNode = grid.GetNodeFromWorldPoint (targetPos);

        //OPEN = the set of nodes to be evaluated
        List<Node> openSet = new List<Node> ();

        //CLOSED = the set of nodes already evaluated
        HashSet<Node> closedSet = new HashSet<Node> ();

        //add the start node to OPEN
        openSet.Add (startNode);

        /* loop
        * CURRENT = node in OPEN with lowest f_cost
        * remove CURRENT from OPEN
        * add CURRENT to CLOSED
        */
        while (openSet.Count > 0) {
            Node currentNode = openSet [0];
            for (int i = 1; i < openSet.Count; i++) {
                if (openSet [i].fCost < currentNode.fCost ||
openSet[i].fCost == currentNode.fCost && openSet[i].hCost <
currentNode.hCost) {
                    currentNode = openSet [i];
                }
            }
            openSet.Remove (currentNode);
            closedSet.Add (currentNode);

            // if CURRENT is the target node, meaning path is
found
```

```

        if (currentNode == targetNode) {
            RetracePath(startNode, targetNode);
        }

        /*
         * otherwise we need to loop through each of the
neighbour node of the current node
         */
        * foreach NEIGHBOUR of the CURRENT node
        * if NEIGHBOUR is not traversable or NEIGHBOUR is
in the CLOSED
        * skip to the next NEIGHBOUR
        */
        foreach (Node neighbour in
grid.GetNeighbour(currentNode)) {
            if(!neighbour.walkable ||
closedSet.Contains(neighbour)){
                continue;
            }

            int newMovementCostToNeighbour = currentNode.gCost
+ GetDistance (currentNode, neighbour);
            if (newMovementCostToNeighbour < neighbour.gCost
|| !openSet.Contains (neighbour)) {
                neighbour.gCost = newMovementCostToNeighbour;
                neighbour.hCost = GetDistance(neighbour,
targetNode);

                neighbour.parent = currentNode;

                if (!openSet.Contains (neighbour)) {
                    openSet.Add (neighbour);
                }
            }
        }
    }

    public void RetracePath(Node startNode, Node endNode){
        List<Node> path = new List<Node> ();
        Node currentNode = endNode;

        while (currentNode != startNode) {
            path.Add (currentNode);
            currentNode = currentNode.parent;
        }

        //path.Reverse ();

        grid.path = path;
    }

    int GetDistance(Node nodeA, Node nodeB){
        int distX = Mathf.Abs (nodeA.gridX - nodeB.gridX);
        int distY = Mathf.Abs (nodeA.gridY - nodeB.gridY);

        return 10 * (distX + distY);
    }

    /* TODO

```

```

        * NEED HELP
        * how to store n in another list and use it as guide for
enemy NPC to move
        *
        */

        // such sandbox, such experiment, such wow
*****
        public Vector2 GetCoordinate(Vector2 startPos, Vector2
targetPos){

            Node startNode = grid.GetNodeFromWorldPoint (startPos);
            Node targetNode = grid.GetNodeFromWorldPoint (targetPos);

            List<Node> path = new List<Node> ();
            Node currentNode = targetNode;

            List<float> listXCoor = new List<float> ();
            List<float> listYCoor = new List<float> ();

            while (currentNode != startNode) {
                path.Add (currentNode);
                currentNode = currentNode.parent;
                if (currentNode == startNode) {
                    path.Add (currentNode);
                    break;
                }
            }

            foreach (Node n in path) {
                listXCoor.Add (n.worldPos.x);
                listYCoor.Add (n.worldPos.y);
            }
            Vector2 moveTo;

            if (player != null) {
                moveTo = new Vector2 (listXCoor [1], listYCoor [1]);
            } else {
                moveTo = new Vector2 (listXCoor [0], listYCoor [0]);
            }

            return moveTo;
        }
}

```



## *Script 2. MoveNPC.cs*

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class MoveNPC : MonoBehaviour {

    Pathfinding path;

    public Transform player, npc;
    public GameObject playerObject;

    public float timeBetweenKeyPress = 0.5f;
    private float delayTime;

    void Awake() {
        path = GetComponent<Pathfinding> ();
    }

    void Update () {
        if(Time.time >= delayTime && (
            Input.GetKeyDown (KeyCode.W) || Input.GetKeyDown
            (KeyCode.A) || Input.GetKeyDown (KeyCode.S) || Input.GetKeyDown
            (KeyCode.D) ||
                Input.GetKeyDown
            (KeyCode.UpArrow) || Input.GetKeyDown
            (KeyCode.LeftArrow) || Input.GetKeyDown
            (KeyCode.DownArrow) || Input.GetKeyDown (KeyCode.RightArrow))
            && npc.position != player.position)) {

            Vector2 newPos = path.GetCoordinate (player.position,
            npc.position);
            npc.position = newPos;
            delayTime = Time.time + timeBetweenKeyPress;
        }
    }
}
```



### Script 3. Grid.cs

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

public class Grid : MonoBehaviour {

    #region Variable ====
    public Transform player;
    public Transform npc;

    //to determine which layer is unwalkable
    public LayerMask unwalkableMask;

    //define the sort of area in the world coordinate that the
    grid is going to cover
    public Vector2 gridWorldSize;

    //to define how much space each individual node are cover
    public float nodeRadius;

    //2D array of node to represent our grid
    Node[,] grid;

    float nodeDiameter;
    int gridSizeX, gridSizeY;
    #endregion

    void Start(){
        nodeDiameter = nodeRadius * 2;
        gridSizeX = Mathf.RoundToInt (gridWorldSize.x /
nodeDiameter);
        gridSizeY = Mathf.RoundToInt (gridWorldSize.y /
nodeDiameter);

        /*
        * Example, node radius = 0.25; gridWorldSize = (10,10);
        * nodeDiameter = 0.25 * 2
        *             = 0.5
        * gridSizeX = Mathf.RoundToInt (gridWorldSize.x /
nodeDiameter);
        *             = Mathf.RoundToInt(10 / 0.5)
        *             = Mathf.RoundToInt(20)
        *             = 20
        *
        * gridSizeY = Mathf.RoundToInt (gridWorldSize.y /
nodeDiameter);
        *             = Mathf.RoundToInt(10 / 0.5)
        *             = Mathf.RoundToInt(20)
        *             = 20
        *
        * it will create grid with size 20x20
        */

        CreateGrid ();
    }
}
```

```

void CreateGrid(){
    grid = new Node[gridSizeX, gridSizeY];

    Vector2 worldBottomLeft = (Vector2)transform.position -
    Vector2.right * gridWorldSize.x / 2 - Vector2.up * gridWorldSize.y
    / 2;

    /*
    * transform.position = get current position
    * vector2.right = (1,0)
    * vector2.up = (0,1)
    *
    * Vector2 worldBottomLeft = (Vector2)transform.position -
    Vector2.right * gridWorldSize.x / 2 - Vector2.up * gridWorldSize.y
    / 2;
    *
    * Example, gridWorldSize = (10,10)
    * Vector2 worldBottomLeft = (0,0) - (1,0) * 10/2 - (0,1)
    * 10/2
    *
    *                               = (0,0) - (5,0) - (0,5)
    *                               = (-5,-5)
    */

    for (int x = 0; x < gridSizeX; x++) {
        for (int y = 0; y < gridSizeY; y++) {

            //getting each point our node will occupy
            Vector2 worldPoint = worldBottomLeft +
            Vector2.right * (x * nodeDiameter + nodeRadius) + Vector2.up * (y
            * nodeDiameter + nodeRadius);
            /*
            * for x = 0, y = 0
            * Vector2 worldPoint = (-5,-5) + (1,0) * ( 0 *
            0.5 + 0.25) + (0,1) * ( 0 * 0.5 + 0.25)
            *
            *                               = (-5,-5) + (1,0) * (0.25)
            + (0,1) * (0.25)
            *
            *                               = (-5,-5) + (-0.25) + (-
            0.25)
            *
            *                               = (-5.25, -5.25)
            */

            //collision check for each point
            bool walkable
            = !(Physics2D.OverlapCircle(worldPoint, nodeRadius,
            unwalkableMask));
            //Physics2D.OverlapCircle(point:, nodeRadius,
            latyermask)

            //populate our grid with node
            grid [x, y] = new Node (walkable, worldPoint, x,
            y);
        }
    }

    public List<Node> GetNeighbour(Node node){
        List<Node> neighbours = new List<Node> ();

```

```

        for (int x = -1; x <= 1; x++) {
            for (int y = -1; y <= 1; y++) {
                if( x == 0 && y == 0 ||
                    x == -1 && y == 1 ||
                    x == 1 && y == 1 ||
                    x == -1 && y == -1 ||
                    x == 1 && y == -1 )
                    continue;

                /*
                 * GetNeighbour
                 * -----
                 * | -1, 1 | 0, 1 | 1, 1 |
                 * | -1, 0 | 0, 0 | 1, 0 |
                 * | -1,-1 | 0,-1 | 1,-1 |
                 * -----
                 */

                int checkX = node.gridX + x;
                int checkY = node.gridY + y;

                /*
                 * node.gridX and node.gridY value is from
                 * " grid [x, y] = new Node (walkable, worldPoint,
x, y); "
                 * in CreateGrid() class
                 * x and y from loop above
                 * example
                 * (node.gridX,node.gridY) = (5,5) ;
                 * (x,y) = (-1,0)
                 *
                 * checkX   = node.gridX + x
                 *           = 5 + (-1)
                 *           = 4
                 *
                 * checkY   = node.gridY + y
                 *           = 5 + 0
                 *           = 5
                 */

                if (checkX >= 0 && checkX < gridSizeX &&
                    checkY >= 0 && checkY < gridSizeY) {
                    neighbours.Add (grid [checkX, checkY]);
                }

                /*
                 * if (checkX >= 0 && checkY < gridSizeX && checkY
>= 0 && checkY < gridSizeY)
                 * gridSizeX = 20, gridSizeY = 20
                 *
                 * checkX = 4 (>=0) && checkY = 5 (<20) -> true
                 * checkY = 5 (>=0) && checkY = 5 (<20) -> true
                 */

```

```

        * if
        *
        */
    }
}

return neighbours;
//return the list
}

public Node GetNodeFromWorldPoint (Vector2 worldPosition){
    float percentX = (worldPosition.x + gridWorldSize.x / 2) /
gridWorldSize.x;
    float percentY = (worldPosition.y + gridWorldSize.y / 2) /
gridWorldSize.y;

    /*
    * Example,
    * gridWorldSize = (10,10)
    * worldPosition = (7,-9)
    *
    * percentX = (7 + 10/2)/10
    *           = (12)/10
    *           = 1.2
    * percentY = (-9 + 10/2)/10
    *           = (-4)/10
    *           = -0.4
    */

    //float percentX = (worldPosition.x - transform.position.x)
/ gridWorldSize.x + 0.5f - (nodeRadius / gridWorldSize.x);
    //float percentY = (worldPosition.y - transform.position.y)
/ gridWorldSize.y + 0.5f - (nodeRadius / gridWorldSize.y);

    /*
    * Example,
    * gridWorldSize = (10,10)
    * worldPosition = (7,-9)
    * transform.position = (-3,5)
    *
    * percentX = (7-(-3))/(10+0.5f)-(0.25/10)
    *           = (10)/(10.5)-(0.025)
    *           = 0.952 - (0.025)
    *           = 0.927
    * percentY = (-9 + 10/2)/10
    *           = (-4)/10
    *           = -0.4
    */

    percentX = Mathf.Clamp01 (percentX);
    percentY = Mathf.Clamp01 (percentY);

    /*
    * Mathf.Clamp01 -> clamp value between 0 and 1
    *
    * percentX = Mathf.Clamp01(percentX)
    *           = Mathf.Clamp01(1.2)
    *           = 1
    */

```

```

        * percentY = Mathf.Clamp01(percentY)
        *           = Mathf.Clamp01(-0.4)
        *           = -0.4
        *
    */

    int x = Mathf.RoundToInt((gridSizeX - 1) * percentX);
    int y = Mathf.RoundToInt((gridSizeY - 1) * percentY);
    return grid [x, y];

    /*
    * int x = Mathf.RoundToInt((gridSizeX - 1) * percentX);
    *           = Mathf.RoundToInt((20-1)*1);
    *           = Mathf.RoundToInt(19)
    *           = 19
    *
    * int y = Mathf.RoundToInt((gridSizeY - 1) * percentY);
    *           = Mathf.RoundToInt((20-1)*(-0.4));
    *           = Mathf.RoundToInt(19*(-0.4));
    *           = Mathf.RoundToInt(-7.6);
    *           = -8
    *
    * grid[x,y] = grid[19,-8];
    */
}

public List<Node> path;

void OnDrawGizmos(){
    Gizmos.DrawWireCube(transform.position, new
Vector2(gridWorldSize.x,gridWorldSize.y));
    if (grid != null) {
        if (player != null) {
            Node playerNode = GetNodeFromWorldPoint
(player.position);
            Node npcNode = GetNodeFromWorldPoint (npc.position);
            foreach (Node n in grid) {
                Gizmos.color = (n.walkable)?Color.green:Color.red;
                if (playerNode == n)
                    Gizmos.color = Color.blue;
                if (npcNode == n)
                    Gizmos.color = Color.magenta;
                if (path != null) {
                    if (path.Contains (n)) {
                        Gizmos.color = Color.black;
                    }
                }
                Gizmos.DrawCube(n.worldPos, Vector2.one *
(nodeDiameter-0.01f));
            }
        }
    }
}
}

```

## Lampiran 4 Kuesioner

### KUESIONER PENGUJIAN APLIKASI VIDEO GAME MAZE 2D "RETURN TO EARTH"

Nama : Rizki Lestari

Pendidikan / Institusi : Politeknik Negeri Malang

#### Petunjuk Pengisian :

Berikanlah tanda silang (X) pada kolom yang disediakan sesuai dengan penilaian anda.

5	4	3	2	1
Sangat Baik	Baik	Cukup	Kurang	Sangat Kurang

No	Pertanyaan	Nilai				
		5	4	3	2	1
Game						
1	Tahapan antar level		✓			
2	Kesulitan tiap level		✓			
3	Pengambilan jalur NPC	✓				
4	Desain Visual			✓		
User						
1	Kemampuan Memainkan Game			✓		

Bagaimana menurut anda tentang jalur yang diambil oleh NPC?

jalur yang diambil Npc bagus dapat menyesuaikan  
langkah si Pemain

Menurut anda, hal apa saja yang perlu dikembangkan dalam game ini?

- Desain karakter musuh lebih di identikan musuh  
- Desain tombol pada menu utama warnanya disesuaikan  
agar lebih jelas / tidak tabrakan warnanya

## KUESIONER PENGUJIAN APLIKASI VIDEO GAME MAZE 2D "RETURN TO EARTH"

Nama : Dovie Yudhawiratama  
Pendidikan / Institusi : Politeknik Negeri Malang

### Petunjuk Pengisian :

Berikanlah tanda silang (X) pada kolom yang disediakan sesuai dengan penilaian anda.

5	4	3	2	1
Sangat Baik	Baik	Cukup	Kurang	Sangat Kurang

No	Pertanyaan	Nilai				
		5	4	3	2	1
Game						
1	Tahapan antar level	✕				
2	Kesulitan tiap level		✕			
3	Pengambilan jalur NPC	✕				
4	Desain Visual	✕				
User						
1	Kemampuan Memainkan Game			✕		

Bagaimana menurut anda tentang jalur yang diambil oleh NPC?

Jalur yang diambil NPC dapat memprediksi langkah yang diambil user sehingga sudah sangat bagus.

Menurut anda, hal apa saja yang perlu dikembangkan dalam game ini?

- Pengendalian karakter agar lebih responsif lagi
- Penambahan level pada game
- Pengembangan untuk versi portable

**KUESIONER PENGUJIAN APLIKASI  
VIDEO GAME MAZE 2D "RETURN TO EARTH"**

Nama : Dhike Almira  
Pendidikan / Institusi : Politeknik Negeri Malang

**Petunjuk Pengisian :**

Berikanlah tanda silang (X) pada kolom yang disediakan sesuai dengan penilaian anda.

5	4	3	2	1
Sangat Baik	Baik	Cukup	Kurang	Sangat Kurang

No	Pertanyaan	Nilai				
		5	4	3	2	1
Game						
1	Tahapan antar level		✓			
2	Kesulitan tiap level		✓			
3	Pengambilan jalur NPC	✓				
4	Desain Visual				✓	
User						
1	Kemampuan Memainkan Game			✓		

Bagaimana menurut anda tentang jalur yang diambil oleh NPC?

Sangat baik, karena dapat menyesuaikan jalur pergerakan player

Menurut anda, hal apa saja yang perlu dikembangkan dalam *game* ini?

Desain game ~~pada~~ khususnya pada button u/ bermain karena agak membingungkan



**KUESIONER PENGUJIAN APLIKASI  
VIDEO GAME MAZE 2D "RETURN TO EARTH"**

Nama : Anggi Kurniawan  
Pendidikan / Institusi : Politeknik negeri Malang

**Petunjuk Pengisian :**

Berikanlah tanda silang (X) pada kolom yang disediakan sesuai dengan penilaian anda.

5	4	3	2	1
Sangat Baik	Baik	Cukup	Kurang	Sangat Kurang

No	Pertanyaan	Nilai				
		5	4	3	2	1
Game						
1	Tahapan antar level		✓			
2	Kesulitan tiap level		✓			
3	Pengambilan jalur NPC	✓				
4	Desain Visual		✓			
User						
1	Kemampuan Memainkan Game			✓		

Bagaimana menurut anda tentang jalur yang diambil oleh NPC?

Bagus, dapat menyesuaikan jalur pemain

Menurut anda, hal apa saja yang perlu dikembangkan dalam game ini?

- Dalam menentukan arah jalan bagi user yang pertama main masih kebingungan. Sama posisi jalurnya
- Desain visual diperbagus lagi
- bisa dikembangkan ke dalam game berbasis mobile

# KUESIONER PENGUJIAN APLIKASI VIDEO GAME MAZE 2D "RETURN TO EARTH"

Nama : Danur dara Pradnya Basuwar  
Pendidikan / Institusi : Politeknik Negeri Malang

## Petunjuk Pengisian :

Berikanlah tanda silang (X) pada kolom yang disediakan sesuai dengan penilaian anda.

5	4	3	2	1
Sangat Baik	Baik	Cukup	Kurang	Sangat Kurang

No	Pertanyaan	Nilai				
		5	4	3	2	1
Game						
1	Tahapan antar level	✓				
2	Kesulitan tiap level		✓			
3	Pengambilan jalur NPC			✓		
4	Desain Visual			✓		
User						
1	Kemampuan Memainkan Game		✓			

Bagaimana menurut anda tentang jalur yang diambil oleh NPC?

menurut saya sudah masuk akal dan sesuai dengan metode

yang diterapkan. Alangkah baiknya diberikan petunjuk lebih banyak lagi tentang NPC tiap level

Menurut anda, hal apa saja yang perlu dikembangkan dalam game ini?

- music = background music

- desain = Ditambah lebih menarik

et

# KUESIONER PENGUJIAN APLIKASI VIDEO GAME MAZE 2D "RETURN TO EARTH"

Nama : Achmad Prayogi  
Pendidikan / Institusi : Polinema

## Petunjuk Pengisian :

Berikanlah tanda silang (X) pada kolom yang disediakan sesuai dengan penilaian anda.

5	4	3	2	1
Sangat Baik	Baik	Cukup	Kurang	Sangat Kurang

No	Pertanyaan	Nilai				
		5	4	3	2	1
Game						
1	Tahapan antar level	✓				
2	Kesulitan tiap level	✓				
3	Pengambilan jalur NPC	✓				
4	Desain Visual		✓			
User						
1	Kemampuan Memainkan Game	✓		✓		

Bagaimana menurut anda tentang jalur yang diambil oleh NPC?

Cukup cerdas dan tepat

Menurut anda, hal apa saja yang perlu dikembangkan dalam *game* ini?

Desain

## KUESIONER PENGUJIAN APLIKASI VIDEO GAME MAZE 2D "RETURN TO EARTH"

Nama : Melinda Ika Zuliyati  
Pendidikan / Institusi : Politeknik Negeri Malang

### Petunjuk Pengisian :

Berikanlah tanda silang (X) pada kolom yang disediakan sesuai dengan penilaian anda.

5	4	3	2	1
Sangat Baik	Baik	Cukup	Kurang	Sangat Kurang

No	Pertanyaan	Nilai				
		5	4	3	2	1
Game						
1	Tahapan antar level	×				
2	Kesulitan tiap level	×				
3	Pengambilan jalur NPC	×				
4	Desain Visual			×		
User						
1	Kemampuan Memainkan Game			×		

Bagaimana menurut anda tentang jalur yang diambil oleh NPC?

Sangat baik karena jika Player Salah dalam  
Melangkah Maka Player Die

Menurut anda, hal apa saja yang perlu dikembangkan dalam game ini?

Untuk Desain Lebih dikembangkan lagi ya...

### KUESIONER PENGUJIAN APLIKASI VIDEO GAME MAZE 2D "RETURN TO EARTH"

Nama : Ersa Pahmaurati  
Pendidikan / Institusi : Politeknik Negeri Malang

**Petunjuk Pengisian :**

Berikanlah tanda silang (X) pada kolom yang disediakan sesuai dengan penilaian anda.

5	4	3	2	1
Sangat Baik	Baik	Cukup	Kurang	Sangat Kurang

No	Pertanyaan	Nilai				
		5	4	3	2	1
Game						
1	Tahapan antar level		X			
2	Kesulitan tiap level	X				
3	Pengambilan jalur NPC		X			
4	Desain Visual		X			
User						
1	Kemampuan Memainkan Game		X			

Bagaimana menurut anda tentang jalur yang diambil oleh NPC?

Sudah baik untuk mencari jalur terdekat

Menurut anda, hal apa saja yang perlu dikembangkan dalam game ini?

Characternya ditambahkan

## KUESIONER PENGUJIAN APLIKASI VIDEO GAME MAZE 2D "RETURN TO EARTH"

Nama : M. Akbar H

Pendidikan / Institusi : POLINEMA

### Petunjuk Pengisian :

Berikanlah tanda silang (X) pada kolom yang disediakan sesuai dengan penilaian anda.

5	4	3	2	1
Sangat Baik	Baik	Cukup	Kurang	Sangat Kurang

No	Pertanyaan	Nilai				
		5	4	3	2	1
Game						
1	Tahapan antar level		X			
2	Kesulitan tiap level		X			
3	Pengambilan jalur NPC				X	
4	Desain Visual			X		
User						
1	Kemampuan Memainkan Game		X			

Bagaimana menurut anda tentang jalur yang diambil oleh NPC?

Jalur ketika Start hingga 2/3 langkah mendekati Player cukup baik, namun ketika NPC berada tepat dibelakang Player jalur yang diambil terkesan monoton

Menurut anda, hal apa saja yang perlu dikembangkan dalam game ini?

Animasi, Desain Portal, Player dan NPC Tiap level dibedakan  
Musik pada Main menu



### KUESIONER PENGUJIAN APLIKASI VIDEO GAME MAZE 2D "RETURN TO EARTH"

Nama : AFRIKA CHORI Q  
Pendidikan / Institusi : POLIDENSA

**Petunjuk Pengisian :**

Berikanlah tanda silang (X) pada kolom yang disediakan sesuai dengan penilaian anda.

5	4	3	2	1
Sangat Baik	Baik	Cukup	Kurang	Sangat Kurang

No	Pertanyaan	Nilai				
		5	4	3	2	1
Game						
1	Tahapan antar level	X				
2	Kesulitan tiap level	X				
3	Pengambilan jalur NPC	X				
4	Desain Visual		X			
User						
1	Kemampuan Memainkan Game		X			

Bagaimana menurut anda tentang jalur yang diambil oleh NPC?

Sebaiknya NPC terus berjalan, tanpa astronaut harus  
berjalan.

Menurut anda, hal apa saja yang perlu dikembangkan dalam game ini?

antarmuka, stor, online, multiplayer, animasi.  
Kumpulkan koin untuk bisa membeli karakter baru.

## Lampiran 5 Profil Penulis



Nama : Ari Mahardika Ahmad Nafis

NIM : 1341180068

Tempat, Tanggal Lahir : Batu-Malang, 06 November 1995

Jenis Kelamin : Laki-Laki

No Hp : 085646777395

Email : arimahardika.an@gmail.com