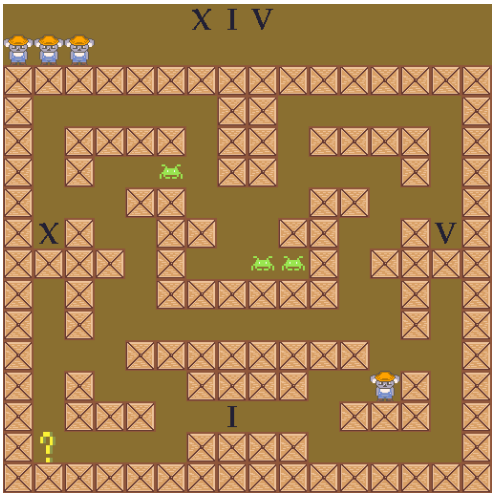
BAB II
TEORI PENUNJANG

1.1 Animasi 2D

Animasi dua dimensi (2D) merupakan penciptaan gambar bergerak di dalam lingkungan 2D pada setiap *frame-by-frame* untuk mensimulasikan gerakan dari gambar[5].

1.1.1 Animasi Cel

Animasi Cel berasal dari kata "*Celluloid*" yaitu bahan dasar dalam pembuatan animasi. Animasi cel merupakan lembaran-lembaran yang membentuk animasi tunggal, masing-masing cel merupakan bagian yang terpisah sebagai objek animasi. misalnya ada tiga buah animasi cel, cel pertama berisi satu animasi karakter, cel kedua berisi animasi karakter lain, dan cel terakhir berisi latar animasi. Ketiga animasi cel ini akan disusun berjajar, sehingga ketika dijalankan animasinya secara bersamaan, terlihat seperti satu kesatuan[5].



Gambar 2.1 Contoh dari prototype game Maze Roman Numerals

1.1.2 Animasi Path

Animasi Path merupakan animasi dari objek yang gerakannya mengikuti garis lintasan yang sudah ditentukan. Biasanya dalam animasi path diberi perulangan animasi, sehingga animasi terus berulang hingga mencapai kondisi tertentu. Animasi jenis ini didapatkan dengan teknik animasi path, teknik ini menggunakan layer tersendiri yang didefinisikan sebagai lintasan gerakan objek [5].

Gambar 2.3 Contoh dari prototype game Maze Roman Numerals

1.2 Aseprite

Aseprite merupakan aplikasi pengembangan *pixel art* yang menciptakan gambar berupa animasi 2D dan bekerja pada operasi sistem Windows, macOS, dan Linux Ubuntu.

1.3 Pixel Art

Pixel Art adalah karya seni digital yang diciptakan oleh lewat perangkat lunak dimana gambar dibuat dalam tingkatan pixel (elemen terkecil dalam sebuah gambar yang ditampilkan dalam layar komputer) [5].

Gambar 2.4. Contoh dari game Space Invaders

1.4 Godot Engine

Godot Engine merupakan mesin game 2D dan 3D, *lintas-platform*, gratis dan *open-source* yang dirilis di bawah lisensi *MIT*. Awalnya dikembangkan untuk beberapa perusahaan di Amerika Latin sebelum dirilis ke publik. Lingkungan pengembangan berjalan pada beberapa sistem operasi termasuk Linux, macOS, dan Windows [6].

1.5 Sprite

adalah gambar bitmap dua dimensi menjadi komponen dalam layar yang lebih besar. Sprite banyak digunakan dalam permainan video dua dimensi untuk menggambarkan objek seperti tokoh karakter yang dapat bergerak. atau benda-benda lainnya. Sebuah sprite dapat diam atau bergerak di layar, dan animasi dapat dicapai dengan mengubah posisi atau berubah menjadi sprite lain sehingga seolah-olah terjadi animasi. Dulu, sprite merupakan komponen yang umum dalam permainan video, dan kini masih digunakan dalam permainan-permainan sederhana, karena teknik ini cukup hemat dan efektif. Namun, teknik ini memiliki keterbatasan karena bentuknya tidak dapat berubah [6].

1.6 Tileset

Tileset adalah sebuah gambar berisi sekumpulan gambar yang mempunyai ukuran masing-masing, misalnya 32x32 pixel bisa disusun sedemikian rupa untuk membangun sebuah gambar besar [6].

Gambar 2.5. Contoh dari prototype game Maze Roman Numerals

1.7 Gd Script

GD Script Merupakan bahasa pemrograman dari Godot engine. Adapun bisa menggunakan bahasa pemrograman lainnya pada Godot engine untuk pengembangan sebuah game yang lebih spesifik[6].

1.8 UML

Unified Modeling Language atau biasa disingkat (UML) membahas mengenai spesifikasi standar untuk mendokumentasikan, spesifikasi, dan membangun perangkat lunak[9].

1.9 Algoritma A Star

Menurut *Arhami* (2006) algoritma A Star merupakan algoritma *best first search* (BFS) dengan pemodifikasian fungsi *heuristik*. Algoritma ini meminimalkan total biaya lintasan, dan pada kondisi yang tepat akan memberikan solusi yang terbaik dalam waktu yang optimal[8].

Secara sistematis, fungsi sebagai estimasi fungsi evaluasi terhadap *node* (simbol *n*) dapat dituliskan dengan persamaan

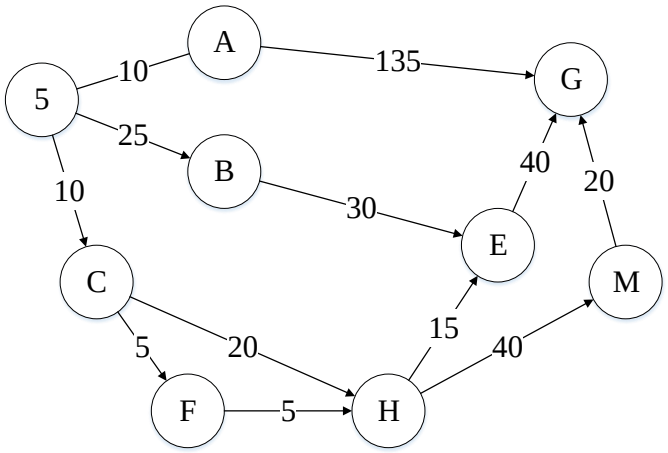
$$F'(n) = g(n) + h(n)$$

Keterangan :

$F'(n)$ = biaya estimasi terendah

$g(n)$ = biaya dari *node* awal ke *node* n

$h(n)$ = perkiraan biaya dari *node* n ke *node* akhir



Gambar 2.6. Simulasi Algoritma A Star

n	S	A	B	C	E	F	G	H	M
h(n)	80	70	70	70	75	78	0	70	70

Tabel 2.1 Tabel Node Simulasi Algoritma A Star

pada penerapan algoritma A Star memiliki terminologi dasar diantaranya *starting point*, simpul (*node*), *A*, *open list*, *closed list*, harga (*cost*), halangan (*unwalkable*).

1. *Starting point* adalah terminologi untuk posisi awal sebuah benda
2. *A* adalah simpul (*node*) yang sedang dijalankan dalam algoritma pencarian jalur terpendek[7].
3. *Node* adalah petak-petak kecil sebagai representasi dari area *pathfinding* bentuknya dapat berupa persegi, lingkaran, maupun segitiga[7].
4. *Open list* adalah tempat menyimpan data simpul (*node*) yang mungkin di akses dari *starting point* maupun *node* yang sedang dijalankan[7].
5. *Closed list* adalah tempat menyimpan data simpul (*node*) sebelum *A* yang juga merupakan bagian dari jalur terpendek yang telah berhasil didapatkan[7].
6. *Cost* adalah nilai yang diperoleh dari penjumlahan, jumlah nilai tiap simpul (*node*) dalam jalur terpendek dari *starting point* ke *A*, dan jumlah nilai perkiraan dari sebuah simpul (*node*) ke simpul (*node*) tujuan[7].

7. Simpul tujuan yaitu simpul (*node*) yang dituju[7].
8. Halangan (*unwalkable*) adalah sebuah atribut yang menyatakan bahwa sebuah simpul (*node*) tidak dapat dilalui oleh *A*[7].

1.10 Upper Confidence Bound

Upeer Confidence Bound atau disingkat UCB merupakan salah satu metode yang mempunyai potensi tinggi dalam penerapan di berbagai permainan dalam game. UCB melakukan maintaining rata-rata perhitungan *payoff*. Perhitungan dilakukan dengan rumus sebagai berikut[1].

$$v(i) = x(i) + \sqrt{\frac{k\ in(t)}{c(i)}}$$

$v(i)$ = *payoff* yang di dapat dari hasi perhitungan *arm*

$x(i)$ = hasil dari penggunaan *arm*

$c(i)$ = berapa kali arm tersebut dijalankan

k = bilangan konstanta

t = waktu ketika menjalankan UCB

Pada keterangan variabel rumus UCB diatas. Variabel *v* merupakan *payoff* yang dipengaruhi oleh nilai konstanta variabel *k*, jika semakin besar nilai variabel *k* maka perpindahan arm akan lebih cepat. Sebaliknya jika nilainya kecil, maka perpindahan antar arm akan lebih lama. Nilai variabel *t* dalam perhitungan UCB akan selalu di tambah setiap kali perhitungan UCB dijalankan [1].

Rock				Paper			Scissors		
Time	*C(I)*	*X(I)*	*V(I)*	*C(I)*	*X(I)*	*V(I)*	*C(I)*	*X(I)*	*V(I)*
0	0	0	∞	0	0	∞	0	0	∞
1	1	0	0.00	0	0	∞	0	0	∞
2	1	0	1.18	1	1	2.18	0	0	∞

3	1	0	1.48	1	1	2.48	1	-1	0.48
4	1	0	1.67	2	1	2.18	1	-1	0.67
5	1	0	1.79	3	1	2.04	1	-1	0.79
6	1	0	1.89	4	1	1.95	1	-1	0.89
7	1	0	1.97	5	1	1.88	1	-1	0.97

Tabel 2.2 Simulasi pada Game Rock Paper Scissors

Pada Tambel diatas adalah hasil simulasi Game Rock Paper Scissors bahwa player akan mendapatkan asumsi terbaik untuk pada *Paper* dari Time ke-2 sampai ke-6, sedangkan Time 1,2,3 UCB akan mencoba semua kemungkinan terlebih dahulu untuk mendapatkan hasil *Payoff* pertama [1].