BAB 2
TINJAUAN PUSTAKA

2.1 Game

Game atau permainan adalah sesuatu yang ditempatkan dalam dunia buatan yang sudah diatur melalui ketentuan – ketentuan (*rules*) yang sudah ditetapkan. Ketentuan tersebut digunakan untuk menentukan semua aksi dan tindakan yang harus dilakukan dan yang tidak oleh pemain dalam permainan. *Game* termasuk dalam bentuk interaktif, partisifatif dan hiburan. Maksudnya orang yang bermain *game* akan terhibur dengan ikut berpartisipasi kedalam *game*nya secara aktif [9].

Berdasarkan representasinya, game dapat dibedakan menjadi 2 jenis yaitu game 2 dimensi (2D) dan 3 dimensi (3D). Game 2D adalah game yang secara matematis hanya melibatkan 2 elemen koordinat kartesius yaitu x dan y, sehingga konsep kamera pada game 2D hanya menentukan gambar pada game yang dapat dilihat oleh pemain.Sedangkan game 3D adalah game yang selain melibatkan elemen x dan y juga melibatkan elemen pada perhitungannya sehingga konsep kamera pada game 3D benar-benar menyerupai konsep kamera pada kehidupan nyata.

2.1.1 Game 2D

Game dua dimensi dapat diketahui berdasarkan ruangnya yang hanya memiliki dua sisi (X dan Y). Sedangkan untuk gambarnya sendiri dapat menggunakan vector maupun Bitmap.Untuk membuat animasi bergerak seperti berjalan, melompat, berlari, dan lainnya maka harus membuat gambar satu persatu yang disebut dengan frame. Kerealisasian gerakan ditentukan dari gambar yang dibuat, jumlah gambar (frame) yang digunakan, serta hitungan gambar per detik (*frame per second*) semakin tinggi hitungan gambar per detik maka semakin mulus gerakan yang akan dihasilkan. Game 2D atau dua dimensi adalah konsep di mana semua objek berada pada satu bidang datar. Pemain game 2D tidak dapat bergerak bebas ke segala sisi, gerakan pada game 2D dibatasi hanya horisontal dan vertikal atau secara koordinat

gerakan dibatasi hanya dapat bergerak pada sumbu X danY. Berdasarkan pergerakannya, game 2D dapat dibagi menjadi :

1. Game statis adalah game yang gambar latar dan tempat game 2D tidak bergerak sama sekali contohnya adalah tetris.
2. *Side Scrolling* adalah game yang saat dimainkan, kamera akan bergeser ke kanan atau ke kiri dengan kecepatan sesuai dengan gerakan dan kecepatan karakter yang dimainkan pada game tersebut, contohnya seperti super Mario bross, Sonic dan Megaman. Pada side scrolling game sendiri dapat juga ditemukan game dimana screen yang tidak bergerak mengikuti karakter game, ada juga jenis game yang harus melewati berbagai rintangan, menembak musuh, mengkoleksi berbagai item seiring dengan screen yang bergerak sebelum screen tersebut terlewati [11].

2.1.2 Game Multiplayer

Game multiplayer adalah jenis game yang dapat dimainkan lebih dari satu pemain pada arena game yang sama dan pada waktu yang bersamaan pula. Setiap pemain dalam game multiplayer memungkinkan setiap pemainnya saling berinteraksi dengan pemain lainnya, bekerja sama dalam tim yang sama, menjadi lawan tanding, hingga mampu menyediakan bentuk komunikasi sosial yang hampir tidak ditemukan pada game dengan orientasi single-player [11].

2.1.3 Genre Game

Berdasarkan genre, game dapat dibagi menjadi beberapa genre yaitu :

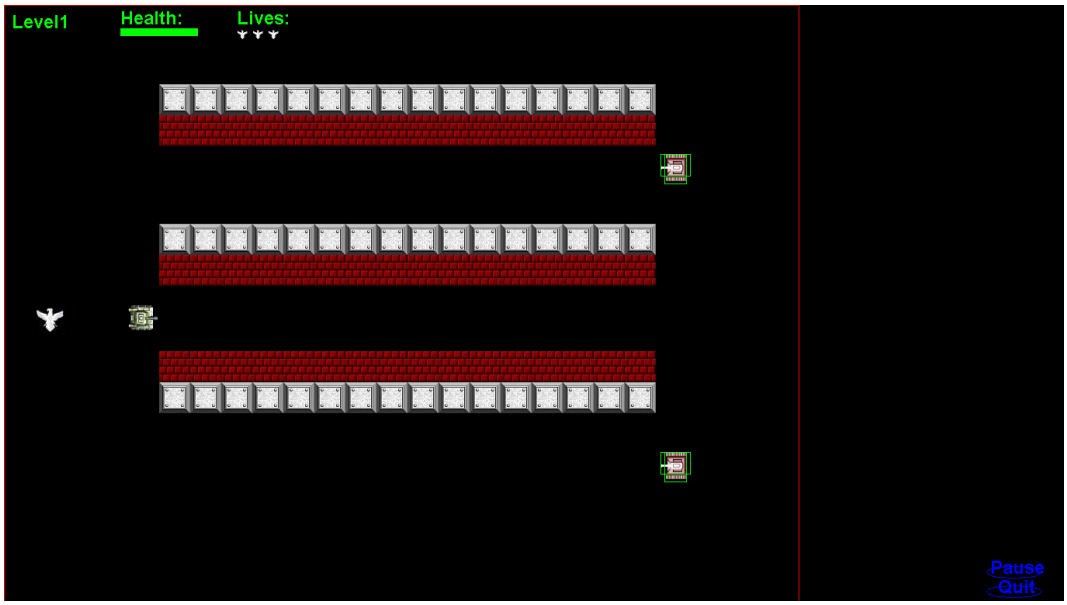
1. *Action Game,* Sebuah game action membutuhkan pemain dengan kecepatan reflex, akurasi, dan ketepatan waktu untuk menghadapi sebuah rintangan. Biasanya mempunyai gameplay yang berhubungan dengan pertarungan. Ada banyak sub jenis dari game action, contohnya fighting games dan first-person shooters.
2. *Adventure Game*, game genre ini dikategorikan sebagai game play yang mengharuskan pemain memecahkan bermacam–macam teka-teki melalui interaksi dengan orang lingkungan dalam game tersebut.

3. *Role Playing Game,* Role playing game adalah game yang memiliki game play dimana karakter milik player akan berpetualang dengan skill combat dalam cerita game.
4. *Simulation Game,* Genre ini bertujuan untuk memberikan pengalaman simulasi kepada pemain.
5. *Strategy Game,* Genre ini berfokus pada gameplay dimana dibutuhkan pemikiran yang tepat agar dapat meraih kemenangan.
    a. *Real Time Action,* Dalam real time action, action dilakukan dalam waktu yang bersamaan oleh masing-masing pihak dimana action dimainkan per ronde atau bergiliran. Contoh game dari genre ini adalah Warcraft series.
    b. *Tactical Game,* Dalam genre ini pemain harus menggunakan bermacam-macam taktik dan action untuk mencapai kemenangan. Contoh game dari genre ini adalah Dark Omen.
    c. 4X Game, Genre ini berarti penjelajahan, menjajah dan memusnahkan. Contoh game dari genre ini adalah Galactic Civizations.
    d. *Artillery Game,* Genre game ini biasanya mengikutkan combat dengan tank atau tentara militer. Contoh game dari genre ini adalah Tanarus.
6. *Vehicle Simulation,* Game Genre ini merupakan simulasi yang memberikan pemain sebuah pengalaman realistik dalam mengendarai kendaraan tertentu [11].

2.2 Game War-Tank

Game War-tank adalah game yang sebenarnya sama dengan game Battle-City akan tetapi beda pada tampilan map, musuh (NPC) diberikan AI (*artificial intelegence*) atau kecerdasan buatan untuk menemukan tank pemain. Pada game ini digambarkan sebuah tank yang dimainkan single player melawan beberapa tank musuh lainnya. Dengan HP (Health Point) yang dimiliki pada setiap tank tersebut menjadikan sebuah acuan tank itu kalah atau menang. Agar NPC musuh dapat memiliki kemampuan menyerang dalam keadaan apapun, maka harus diberikan kecerdasan buatan pada NPC musuh tersebut.

War-Tank merupakan salah satu permainan klasik, dimana game ini hanya memiliki sebuah tampilan jendela tanpa dapat di scroll. Pemain harus dapat mempertahankan markas dari seluruh serangan musuh dan menghancurkan setiap musuh yang ada. Tampilan game War-Tank dapat dilihat pada Gambar 2.1 berikut.



Gambar 2.1 Tampilan di dalam Game War-Tank

Seluruh area pada permainan ini ditutupi oleh berbagai macam layout, seperti batu bata dan sebagainya. Batu bata dapat hancur dengan sekali tembak, dan satu satunya bahan yang digunakan untuk melindungi markas dari tank musuh. Macam-macam layout dapatkan dilihat pada gambar di bawah ini:

Brick (Batu Bata), dapat di hancurkan dengan tembakan tank. Lihat Gambar 2.2.

Gambar 2.2 Brick

Base/Home, Markas yang harus ditembaki oleh Musuh. Lihat Gambar 2.3.

Gambar 2.3 Base/Home

Metal (Besi), tidak dapat dihancurkan. Lihat Gambar 2.4.

Gambar 2.4 Metal

Contoh Tank yang digunakan oleh player. Lihat Gambar 2.5.

Gambar 2.5 Tank Player

Contoh Tank yang digunakan oleh musuh. Lihat Gambar 2.6.

Gambar 2.6 Tank musuh

2.3 Artificial Intelligence

Kecerdasan buatan atau *Artificial Intelligence* (AI) merupakan cabang dari ilmu komputer yang berhubungan dengan pengotomatisan tingkah laku cerdas. Andreas Kaplan dan Michael Haenlein mendefinisikan kecerdasan buatan sebagai "kemampuan sistem untuk menafsirkan data eksternal dengan benar, untuk belajar dari data tersebut, dan menggunakan pembelajaran tersebut guna mencapai tujuan dan tugas tertentu melalui adaptasi yang fleksibel"[4]. Kecerdasan Buatan peran penting di era kini dan masa yang akan dating. Bidang ini telah berkembang sangat pesat di 20 tahun terakhir seiring dengan pertumbuhan kebutuhan akan perangkat cerdas pada industri dan rumah tangga [4].

Berdasarkan sudut pandang, kecerdasan buatan dapat dipandang sebagai berikut :

1. Sudut pandang kecerdasan, adalah bagaimana membuat mesin yang cerdas dan dapat melakukan hal-hal yang sebelumnya hanya dapat dilakukan manusia.

2. Sudut pandang bisnis, kecerdasan buatan adalah sekelompok alat bantu yang berdayaguna dan metodologi yang menggunakan alat-alat bantu tersebut untuk menyelesaikan masalah-masalah bisnis.
3. Sudut pandang pemrograman, kecerdasan buatan meliputi studi tentang pemrograman simbolik, pemecahan masalah dan proses pencarian.
4. Sudut pandang penelitian :
    a. Riset tentang kecerdasan buatan dimulai pada awal tahun 1960-an, percobaan pertama adalah membuat program permainan catur,membuktikan teori dan general problem solving.
    b. Kecerdasan buatan adalah nama pada akar dari studi area.

Kecerdasan buatan memiliki sejumlah sub disiplin ilmu yang sering digunakan untuk pendekatan yang esensial bagi penyelesaian suatu masalah dan dengan aplikasi bidang kecerdasan buatan yang berbeda. Aplikasi penggunaan kecerdasan buatan dapat dibagi ke dalam tiga kelompok yaitu :

1. Expert Task

   Kecerdasan buatan dibentuk berdasarkan pengalaman dan pengetahuan yang dimiliki oleh para ahli yang bertujuan untuk membantu para ahli dalam menyampaikan ilmu-ilmu yang dimiliki.Contohnya adalah Analisis finansial, Analisis medical, Analisis ilmu pengetahuan, Rekayasa (desain,pencarian,kegagalan,perencanaan,manufaktur).
2. Formal Task

   Kecerdasan buatan digunakan untuk melakukan tugas-tugas formal yang selama ini manusia biasa lakukan dengan lebih baik. Contohnya adalah Game, Matematika (geometri,logika,kalkulus,integral).
3. Mundane Task

   Secara harfiah mundane adalah keduniaan. Kecerdasan buatan digunakan untuk melalukan hal-hal yang sifatnya duniawi atau melakukan kegiatan yang dapat membantu manusia. Contohnya adalah Persepsi, Bahasa alami, Robot control.

   Aplikasi kecerdasan buatan memiliki dua bagian utama yaitu :

1. Basis Pengetahuan (Knowledge Base) berisi fakta-fakta, teori, pemikiran dan hubungan antara satu dengan lainnya.
2. Motor Inferensi (Inference Engine) kemampuan menarik kesimpulan berdasarikan pengalaman [7].

2.4 Algoritma Harmony Search

Algoritma Harmony Search (HS) pertama kali diperkenalkan oleh Zong Woo Geem pada tahun 2001. Ide dasar algoritma HS adalah meniru proses perbaikan harmoni musik yang dilakukan oleh kelompok paduan musik. Ketika kelompok paduan musik melakukan perbaikan pada harmoni musik yang dimainkan, maka akan terdapat tiga kemungkinan pilihan, antara lain memainkan harmoni musik yang terkenal berdasarkan ingatan mereka, memainkan harmoni musik yang serupa dengan harmoni musik yang terkenal namun ada sedikit penyesuaian, atau membuat harmoni musik yang baru. Geem (2001) memformulasikan ketiga pilihan ini pada proses optimasi secara kuantitatif. Ketiga komponen tersebut diformulasikan menjadi penggunaan harmony memory, penyesuaian nada, dan proses pembangkitan secara random [1].

Penggunaan harmony memory sangat penting karena harmony memory tersebut bisa menjamin bahwa harmoni yang bagus akan dipertimbangkan sebagai elemen-elemen dari vektor solusi yang baru. Agar harmony memory dapat digunakan secara efektif, algoritma HS mengadopsi sebuah parameter yang disebut *Harmony Memory Considering Rate* (HMCR). Jika rate ini terlalu rendah, maka hanya sedikit harmoni elit yang terpilih dan juga dapat menyebabkan proses konvergensi terlalu lambat. Jika rate ini terlalu besar, maka akan menyebabkan nada-nada pada harmony memory banyak terpakai dan tidak sempat mengeksplorasi nada lain, dimana pada akhirnya sulit mencapai solusi yang bagus. Oleh karena itu, biasanya digunakan HMCR = 0.7~0.95 [1].

Komponen kedua adalah penyesuaian nada dimana mempunyai beberapa parameter seperti *bandwidth (bw)* dan *Pitch Adjusting Rate* (PAR). Penyesuaian nada musik berarti pengubahan frekuensi nada, hal itu berarti membangkitkan nilai

yang sedikit berbeda pada algoritma HS. Berikut ini adalah formulasi penyesuaian nada: $^{X}new = {}^{X}old + bw \; x \; \varepsilon$……..……. (2.1)

dimana ^{X}new = nada baru setelah dilakukan penyesuaian nada

^{X}old = nada yang tersimpan pada harmony memory

bw = bandwidth

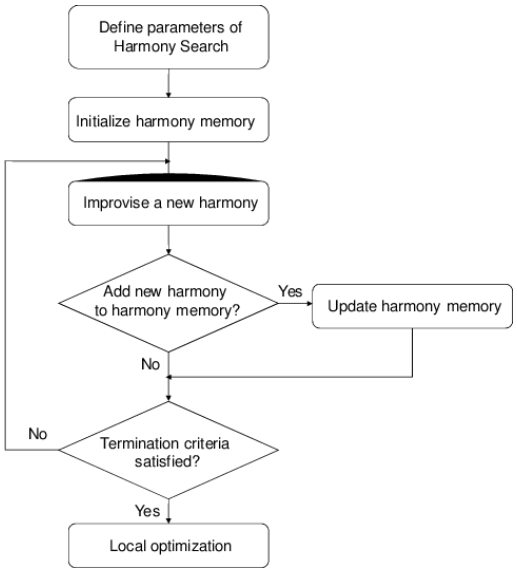
ε = bilangan random dengan interval [-1,1]

PAR yang bernilai rendah dengan bandwidth yang sempit dapat menyebabkan proses konvergensi lambat dikarenakan keterbatasan eksplorasi pada ruang pencarian yang besar. Pada sisi lain, PAR yang tinggi dengan bandwidth yang lebar dapat menyebabkan solusi-solusi yang ada terlalu menyebar dari potensi solusi optimal. Oleh karena itu, biasanya digunakan PAR 0.1~0.5. Berikut ini adalah langkah-langkah dari algoritma HS:

1. Inisialisasi masalah dan parameter algoritma.
2. Inisialisasi harmony memory.
3. Membangkitkan vektor solusi yang baru.
4. Meng-update harmony memory.
5. Mengecek kriteria pemberhentian.[1]

Algoritma Harmony Search (HSA) adalah algoritma yang proses algoritmanya terinsprisasi dari Particle Swarm Optimization (PSO). Particle swarm optimization (PSO) didesain dan dikenalkan oleh Eberhart dan Kennedy. PSO merupakan algoritma pencarian berbasis populasi yang berdasar pada simulasi kelakuan dari burung-burung, lebah atau sekumpulan ikan. Algoritma ini pada dasarnya dimaksudkan untuk mensimulasikan secara nyata pergerakan sekumpulan burung yang bagus dan sukar diprediksi [4].

Berikut *flowchart* untuk Algoritma *Harmony Search*. Lihat pada gambar 2.7.



Gambar 2.7 *Flowchart* Algoritma Harmony Search

2.5 Pengembangan Berorientasi Objek (OOP)

Pengembangan berorientasi objek merupakan suatu pengembangan perangkat lunak yang mengorganisasikan perangkat lunak sebagai kumpulan objek yang berisi data dan operasi di dalamnya. Pengembangan berorientasi objek merupakan suatu cara bagaimana sistem perangkat lunak dibangun melalui pendekatan objek secara sistematis. Pengembangan ini terdiri dari rangkaian aktivitas analisis berorientasi objek, perancangan berorientasi objek, pemrograman berorientasi objek, dan pengujian berorientasi objek.

Pengembangan berorientasi objek memiliki keuntungan sebagai berikut:

1. Pendekatan berorientasi objek dapat dipakai ulang untuk masalah lainnya yang melibatkan objek tersebut atau disebut penggunaan ulang (*reuse*) sehingga dapat meningkatkan produktivitas.
2. Pendekatan berorientasi objek mudah dalam pemeliharaan karena dengan model objek, pola-pola yang cenderung tetap dan stabil dapat dipisahkan dengan pola-pola yang mungkin sering berubah-ubah.

3. Relatif cepat dalam pengembangannya karena sistem yang dibangun dengan baik dan benar pada saat analisis dan perancangan akan menyebabkan berkurangnya kesalahan pada saat pengkodean [8].

Pemrograman berorientasi objek dibagi menjadi beberapa ciri utama, berikut adalah ciri-cirinya :

1. Kelas

Kelas (*class*) adalah contoh abstrak dari sebuah objek yang telah terbentuk dari proses penyederhanaan, dengan kata lain kelas (class) adalah berasal dari objek (*object*), contoh nyata dari sebuah objek dinamakan *instance*. Maka apabila kita mempunyai sebuah kelas manusia, maka beberapa *instances* (wujud nyata) dari kelas manusia adalah Ary, Erik, Rangga dan masih banyak yang lainnya [8]. Perbedaan antara kelas (*class*) dengan objek (object) dalam OOP dibagi menjadi dua, yaitu :

a. *Class* adalah rancangan dan *object* adalah perwujudan dari suatu *class*.
b. *Class* bersifat abstrak sementara object bersifat konkrit (atau nyata).

2. Objek

Sebuah objek pada pemrograman berorientasi objek dapat didefinisikan sebagai berikut :

a. Setiap objek dimiliki oleh kelas objek, sehingga sebuah objek tidak bisa hadir tanpa sebuah kelas yang mendefinisikannya.
b. Sebuah objek adalah sebuah pengkapsulan (*encapsulation*) yang memasukkan data dan operasi untuk pemrosesannya.
c. Atribut-atribut objek membantu untuk menyimpan dan menjaga status objek. Atribut-atribut ini menentukan apa yang berkaitan mengenai objek. *Method* objek adalah satu-satunya cara untuk mengakses data dan memodifikasi statusnya. Cara pengaksesan dan pemodifikasian data dilakukan dengan mengirimkan sebuah pesan ke objek tersebut[8].

2.6 Unified Modeling Language (UML)

Unified modeling language (UML) merupakan model berorientasi objek yang digunakan untuk menggambarkan, menspesifikasi, mendokumentasi dari kontruksi sistem dan pembangunan sistem perangkat lunak. UML merupakan bahasa visual untuk pemodelan dan komunikasi mengenai sebuah sistem dengan menggunakan diagram dan teks-teks pendukungAwal pengembangan UML yaitu pada tahun 1994 dengan melakukan kerja sama antara Grady Booch dan James Rumbaugh dalam mengkobinasi dua metode booch dan OMT. Dan pada tahun 1997, UML di usulkan kepada OMG (*Object Management Group*) yaitu standarisasi teknologi objek supaya dijadikan notasi dan bahasa pemodelan.

Model UML bertujuan untuk :

1. Penyedia bahasa pemodelan grafis yang ekpresif yang siap pakai untuk pengembangan sistem.
2. Penyedia mekanisme dan spesifikasi mendapat konsep - konsep pada sistem
3. Penyedia basis formal untuk pemahaman pemodelan sistem.
4. Mendukung konsep-konsep pengembangan derajat yang lebih tinggi seperti framework, pattern, kolaborasi dan komponen.

UML mengekspresikan pemodelan yang berorientasi objek dalam bentuk diagram yang terdiri dari 13 macam diagram yang dikelompokan dalam 3 kategori. Berikut kategori dan macam-macam diagram tersebut :

1. Kategori *structure diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan suatu struktur statis dari sistem yang dimodelkan. Diagram yang termasuk dalam kategori ini adalah *class diagram, objek diagram, component diagram, compenent struture diagram, package diagram* dan *deployment diagram.*
2. Kategori *behavior diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan kelakuan sistem atau rangkaian perubahan yang terjadi pada sebuah sistem. Diagram yang termasuk dalam kategori ini adalah *use case diagram, activity diagram* dan *state machine diagram*.

3. Kategori *interaction diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan interaksi sistem dengan sistem lain maupun interaksi antar subsistem pada suatu sistem. Diagram yang termasuk kategori ini adalah *sequence diagram,communication digram, timing diagram* dan *interaction overview diagram*[9].

2.6.1 Use case Diagram

Use case diagram menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. *use case* merepresentasikan hubungan antara aktor dengan sistem. *Use case* sebagai berbagai pekerjaan pada sistem sedangkan aktor sebagai sebuah entitas manusia atau mesin yang berinteraksi dengan sistem untuk melakukan pekerjaan-pekerjaan tertentu. *Use case diagram* dapat sangat membantu menyusun kebutuhan sebuah sistem, menjelaskan rancangan dengan klien.

Sebuah *use case* dapat meng-*include* fungsionalitas *use case* lain sebagai bagian dari proses dalam dirinya. *Use case* dapat di-*include* oleh lebih dari satu use case lain, sehingga dapat terhindar dari duplikasi fungsionalitas. *Use case* juga dapat meng-*extend use case* lain dengan behaviournya sendiri. Sementara hubungan generalisasi antar *use case* menunjukkan bahwa *use case* yang satu merupakan spesialisasi dari yang lain [9].

2.6.2 Activity Diagram

Activity diagram menggambarkan alur aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alur berawal, proses apa saja yang dilakukan, dan bagaimana alurnya berakhir. *Activity diagram* menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi. *Activity diagram*
lebih menggambarkan proses-proses dan jalur-jalur aktivitas dari level atas secara umum [9].

2.6.3 Class Diagram

Class Diagram menggambarkan struktur dan deskripsi pada setiap *class, package* dan objek serta hubungan satu sama lain seperti agregasi, pewarisan, asosiasi, dan lain-lain [9].

2.6.4 Sequence Diagram

Sequence diagram menggambarkan hubungan antar objek yang berada di dalam sistem. *Sequence diagram* terdiri atar dimensi vertikal (waktu) dan dimensi horizontal (objek-objek yang terkait). *Sequence diagram* digunakan untuk menggambarkan rangkaian langkah-langkah yang dilakukan sebagai respons dari sebuah *event* untuk menghasilkan keluaran tertentu. Diawali dari apa yang men-*trigger* aktivitas tersebut, proses dan perubahan apa saja yang terjadi secara internal dan keluaran apa yang dihasilkan [9].

2.7 Java

Java merupakan bahasa pemrograman berorientasi objek , yaitu pemodelan yang berdasarkan objek di dunia nyata. Objek sendiri didefinisikan sebagai konsep atau benda yang memiliki batasan, identitas dan arti yang berada dilingkungan sekitar. Java awalnya dibuat ketika perusahaan Sun Microsystem mengerajakan *Green Project* pada tahun 1991. Dan dirilis pada tahun 1995 dengan versi 1.0.2. Nama java sendiri diambil dari kopi kesukaan Gosling yang digiling langsung dari biji kopinya. Java memiliki karakteristik yang sederhana, interpreter, aman, terdistribusi multithreaded dan dinamis. Bahasa pemrograman berorientasi objek memiliki tiga karakteristik utama, yaitu :

1. *Encapsulation*

   Dalam bahasa indonesia disebut pengkapsulan yaitu dasar untuk membatasi ruang linkup program pada data yang diproses. Data dan prosedure dikemas dalam satuan objek sehingga dapat terlindung dari objek luar.

2. *Inheritance*

   Inheritance (pewarisan) diartikan konsep sebuah objek dapat diturunkan data dan prosedur yang dimilikinya ke anak dari objek tersebut. Pewarisan dapat

dilakukan dari kelas induk ke kelas anaknya jika terdapat hubungan antar keduanya. Konsep ini dapat menyederhanakan dan mempermudah dalam program java.

3. *Polymorphism*

   Polimorphism merupakan konsep dari sesuatu yang sama memiliki bentuk dan perilaku yang beda. Dapat diartikan juga penggunaan lebih dari satu metode dengan nama sama. Dengan konsep ini dapat menghindari duplikasi objek [8].

2.8 Manhattan Distance

Manhattan Distance/City Block Distance, merupakan salah satu teknik yang sering digunakan untuk menentukan kesamaan antara dua buah obyek. Pengukuran ini dihasilkan berdasarkan penjumlahan jarak selisih antara dua buah obyek dan hasil yang didapatkan dari *Manhattan Distanc*e bernilai mutlak [14]. Dimana, *Manhattan Distance* melakukan perhitungan jarak dengan cara tegak lurus. Rumus yang digunakan adalah sebagai berikut:

$$\mathrm{D} = \sum_{i=1}^{n} |(xi - xj) + (yi - yj)| \qquad (2.2)$$

Dimana:

D = jarak n = banyak data

i = setiap data x,y = titik awal/yang dituju

2.9 Studi Literatur

Dalam penelitian ini penulis memaparkan beberapa penelitian yang relevan dengan permasalahan yang akan diteliti.

1. Dari penelitian Ade Leonardo dan kawan – kawan, penerapan Logika Fuzzy Mamdani yang digunakan untuk menyerang tank.

   Masalah :

   Permainan Battle Tank merupakan sebuah permainan pertempuran kontak senjata antara dua tank atau lebih pihak. Masing-masing pihak bertujuan untuk mengalahkan pihak yang lain dalam satu arena.

Umumnya permainan Battle Tank penyerangan dilakukan secara bergantian dan ketika HP berkurang maka akan berpengaruh pada tembakan dan menyebabkan tank sulit mengalahkan lawannya.

Solusi :

Oleh karena itu, tank pada permainan Battle Tank membutuhkan kecerdasan buatan yang dapat membantu tank menyerang dengan baik. Jenis kecerdasan buatan yang digunakan untuk memenuhi kebutuhan pada tank yaitu Logika Fuzzy.

Kelebihan :

Logika Fuzzy Mamdani berhasil diterapkan pada game Battle Tank.

Kekurangan :

Mungkin dalam pengembangan penelitian dan permainan, disarankan menambahkan input baru untuk Fuzzy. Misalnya ketika HP sedang lemah atau jarak antara tank musuh dan pemain dekat, maka tank musuh dapat menghindar atau melarikan diri.

Hasil Penelitian :

Berdasarkan hasil implementasi dan evaluasi dari penerapan Logika Fuzzy Mamdani, maka dapat diambil kesimpulan bahwa penerapan Logika Fuzzy metode Mamdani yang digunakan untuk tank dalam permainan Battle Tank memiliki tingkat keberhasilan sebesar 97.7%. Tank dapat menghancurkan tank lainnya dengan cara menembak berdasarkan jarak dan kekuatan tembak yang kemudian menghasilkan damage yang berpengaruh pada pengurangan HP tank lainnya.

2. Menurut penelitian yang ditulis oleh Ditya Timur Aska, Giri Wahyu Wiriasto, dan I Made Budi Suksmadana dapat disimpulkan sebagai berikut :

Masalah :

Pada game “Tank Tempur” digambarkan sebuah tank yang dimainkan *single playe*r melawan beberapa tank musuh lainnya. Dengan HP (*Health Point*) yang dimiliki pada setiap tank tersebut menjadikan

sebuah acuan tank itu kalah atau menang. Agar NPC (*Non-Player Character*) musuh dapat memiliki kemampuan menyerang dalam keadaan apapun, maka harus diberikan kecerdasan buatan pada NPC musuh tersebut.

Solusi :

Fuzzy logic merupakan kecerdasan buatan yang dapat membuat komputer menyelesaikan masalah perilaku sistem yang kompleks dan ketidakpastian yang disebabkan oleh kemampuan mengolah informasi numerik dari variabel yang di ukur. Logika fuzzy dapat diterapkan untuk pengambilan keputusan perilaku komputer dalam game.

Kelebihan :

Penggunaan logika fuzzy metode Tsukamoto pada NPC game tank tempur dapat memberikan abilities (kemampuan) untuk melakukan beberapa opsi keputusan seperti memutuskan terus menyerang, bertahan atau melarikan diri sesuai dengan beberapa kondisi dan pertimbangan yang diberikan.

Kekurangan :

Tujuan dari game yang kurang jelas.

Hasil Penelitian :

Hasil pengujian yang dilakukan, error yang didapat dari nilai absolut aplikasi dikurang dengan nilai matlab dibagi absolut sebesar 3,02%. Sehingga dapat ditarik kesimpulan bahwa penerapan Logika Fuzzy Tsukamoto yang digunakan pada tank NPC berfungsi secara baik [2].

3. Menurut penelitian yang ditulis oleh M. Rio Attoriq dan kawan – kawan menyimpulkan hasil penelitian sebagai berikut :

Masalah :

Pada game *Battle City* tank musuh cukup banyak tapi memiliki pola perilaku serangan yang hampir sama pada setiap tank dikarenakan game tersebut belum menggunakan *Artificial Intelegent* (AI), dan NPC tidak

dapat mencari Base dengan cepat. Oleh karena itu, game harus dilengkapi dengan algoritma.

Solusi :

Algoritma yang akan digunakan untuk melakukan itu adalah algoritma fuzzy logic dan algoritma A*. Logika Fuzzy adalah suatu cara yang tepat untuk memetakan suatu ruang input ke dalam ruang output.

Kelebihan :

Algoritma Fuzzy Logic Sugeno dan A* dapat diterapkan pada game Battle City.

Kekurangan :

Penelitian menggunakan 1 NPC mungkin nantinya NPC bisa ditambahkan kembali.

Hasil Penelitian :

Hasil uji dari 30 data sampel uji algoritma fuzzy logic sugeno menunjukan bahwa tingkat keberhasilan menentukan prilaku NPC sebesar 100%. Hasil uji dari 30 data sampel uji algoritma A* menunjukan bahwa tingkat keberhasilan mencari jalur terpendek sebesar 100% pada permainan Battle City.

4. Penelitian mengenai Algoritma Harmony Search di implementasikan pada RTS Game oleh Herti Miawarni disebutkan sebagai berikut :

Masalah :

Beberapa Penelitian tentang perilaku NPC yang telah dilakukan, diantaranya adalah optimisasi strategi. Adapun keterkaitan penelitian ini dengan penelitian sebelumnya adalah pada permasalahan daftar penugasan (Assignment List) yang merupakan bagian dari strategi dalam game.

Solusi :

Pada penelitian ini, dilakukan uji coba algoritma optimisasi HSA untuk menghasilkan daftar penugasan bagi tiap-tiap NPC. Sehingga

pergerakan dan perilaku menyerang tiap NPC akan lebih terkoordinir dan optimal terutama dari segi total tempuh.

Kelebihan :

Kinerja algoritma HSA dalam meminimumkan rata-rata total tempuh sangat baik. Hal ini ditunjukkan dari hasil rata-rata total tempuh terpendek sebesar 384.12 pada saat menggunakan mode HSA vs HSA. Sedangkan mode HSA vs Non Optimisasi mencapai nilai rata-rata total tempuh sebesar 403.12. Dan rata-rata total tempuh terjauh didapat dari mode Non Optimisasi vs Non Optimisasi yaitu sebesar 451.68.

Kekurangan :

Waktu komputasi pada mode HSA lebih lama dibanding Non Optimisasi. Dari 10 kali trial, algoritma HSA memiliki rata-rata waktu komputasi penugasan 0.69 detik dan Non Optimisasi memiliki waktu komputasi lebih singkat yaitu 0.23 detik.

Hasil Penelitian :

Dari uji coba yang dilakukan, perilaku pergerakan menyerang paling realistik diperoleh pada saat kedua kubu NPC saling menggunakan HSA, dengan waktu komputasi sebesar 0.69 detik, dan rata-rata total tempuh mencapai nilai terpendek sebesar 384.12 [1].

5. Dari penelitian Irinne Puspitasari dan Purwanto, penerapan yang dilakukan terhadap beberapa contoh kasus VRPTW yang diberikan yakni:

Masalah :

Masalah pendistribusian yang sering digunakan oleh masyarakat dewasa ini adalah permasalahan mengenai Vehicle Routing Problem (VRP). Tujuan yang ingin dicapai adalah mencari biaya transportasi minimum dengan armada kendaraan yang memiliki kapasitas yang bervariasi dengan tujuan pengiriman yang bervariasi sesuai dengan permintaan pelanggan.

Solusi :

Pada penelitian ini akan diidentifikasi langkah-langkah algoritma Harmony Search untuk VRPTW secara rinci dan diterapkan dalam beberapa contoh kasus yang selanjutnya akan dibandingkan dengan algoritma Tabu Search.

Kelebihan :

Harmony Search bisa diterapkan pada kasus VRPTW.

Kekurangan :

Harmony Search belum memberikan rute solusi dengan waktu pelayanan lebih pendek jika dibandingkan dengan algoritma Tabu Search. Penyebab dari hal tersebut adalah proses iterasi yang singkat. Akan tetapi dengan adanya penambahan fungsi obyektif pada Harmony Search mampu memberikan hasil yakni total jarak tempuh yang lebih baik dari Tabu Search.

Hasil Penelitian :

Penerapan yang dilakukan terhadap beberapa contoh kasus VRPTW yang diberikan yakni contoh kasus dengan 8 titik dan 10 titik, Harmony Search mampu memberikan hasil akhir rute yang lebih beragam sehingga kemungkinan menemukan solusi optimal akan lebih besar, akan tetapi tidak mampu memberikan jaminan akan menemukan rute akhir yang paling optimum dengan kata lain tidak mampu menjamin penemuan global optimum.