BAB 2

TINJAUAN PUSTAKA

2.1 Teori Umum

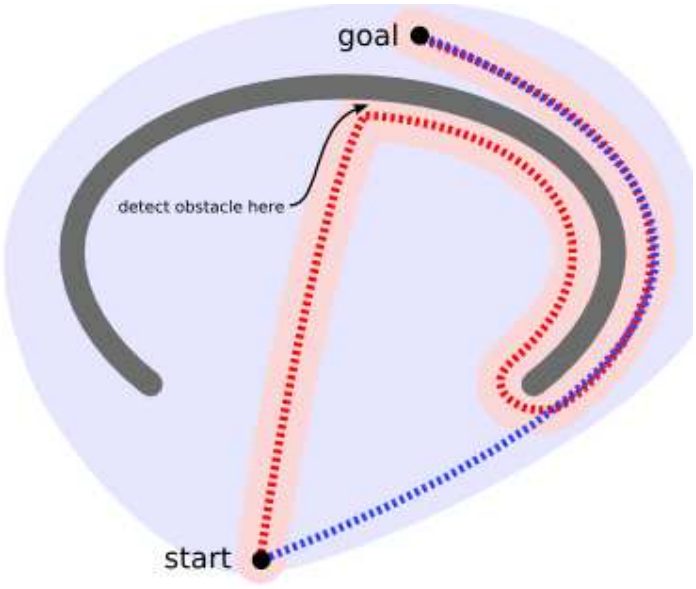
2.1.1 *Traveling Salesman Problem*

Berdasarkan contoh yang diperoleh dari Thomas H.Cormen, Charles E. Leiserson, Ronald L. Rivest, dan Clifford Stein (2009), *Traveling Salesman Problem* adalah teori bagaimana sebuah jalur yang dilewati oleh sebuah objek adalah jalur terpendek yang dapat dicapai dari semua kemungkinan jalur yang dapat diambil.

2.1.1.1A* (A *Star*)

Metode *pathfinding* yang digunakan pada *game* adalah algoritma A*. A* mudah untuk diimplementasikan dan sangat efisien dan memiliki banyak ruang lingkup untuk dioptimalkan. A* merupakan metode yang dapat digunakan didalam pathfinding. Dengan menggunakan algoritma A*, kita dapat menemukan jalur terpendek. Algoritma ini membuang langkah-langkah yang tidak perlu dengan pertimbangan bahwa langkah-langkah yang dibuang sudah dipastikan tidak akan mencapai solusi yang diinginkan.

Prinsip algoritma ini adalah mencari jalur terpendek dari sebuah simpul awal (*Starting Point*) menuju simpul tujuan dengan memperhatikan harga (F) terkecil. A* memperhitungkan *cost* dari *current state* ke tujuan dengan fungsi heuristik, algoritma ini juga mempertimbangkan *cost* yang telah ditempuh selama ini dari *initial state* ke *current state*. Jadi jika ada jalan yang telah ditempuh sudah terlalu panjang dan ada jalan lain yang *cost* nya lebih kecil tetapi memberikan posisi yang sama jika dilihat dari *goal*, maka jalan yang lebih pendek yang akan dipilih.



Gambar 2.1 A*

Metode ini digunakan karena A* bekerja dengan cara mencari jalur terpendek secara keseluruhan, bukan hanya melihat jalur terkecil berdasarkan nodenya berikutnya saja, sehingga lebih efektif digunakan pada area dengan banyak *obstacle* didalamnya.

2.1.2 *Decision Making*

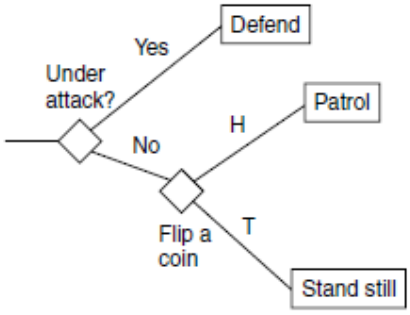
Decision Making adalah serangkaian algoritma yang dirancang dengan memasukan beberapa kemungkinan langkah yang bisa diambil oleh suatu aplikasi, Pada *game* ini *decision making* memberikan kemampuan suatu karakter untuk menentukan langkah apa yang akan diambil. *Decision making* dilakukan dengan cara menentukan satu pilihan dari list yang sudah dibuat pada algoritma yang dirancang.

Algoritma *decision making* kerap digunakan dalam aplikasi *game*, akan tetapi algoritma *decision making* dapat diimplementasikan pada banyak aplikasi lain.

2.1.2.1 *Random Decision Trees*

Random Decision Trees adalah algoritma yang membentuk serangkaian langkah-langkah yang akan dimasukan kedalam algoritma *decision trees*. Setiap pilihan langkah yang dimasukkan pada *decision trees* tidak dapat diprediksi, setiap langkah akan dilakukan secara acak berdasarkan nilainya.

Random Decision Trees pada *game "Academy Story"* akan bekerja dengan beberapa pilihan, jika karakter sedang tidak melakukan kegiatan maka *random decision trees* akan bekerja berdasarkan nilai *random* yang telah ditentukan, apabila karakter sedang berada didalam kelas atau sedang melakukan kegiatan maka *random decision trees* tidak dilakukan dan karakter akan menyelesaikan tindakan nya terlebih dahulu.



Gambar 2.2 *Random Decision Trees*

Metode ini digunakan karena tingkat kerumitannya cukup mudah, gampang diimplementasikan dan sudah mencakup ruang lingkup *game*.

*2.1.3 PBO (*Pemrograman Berorientasi Objek*)*

Menurut Matt Weisfeld (2009), Pemrograman Berorientasi Objek adalah sebuah data dan operasi-operasinya yang dapat memanipulasi data yang sudah dienkapsulasi didalam sebuah objek. Didalam pemrograman berorientasi objek ada beberapa konsep sebagai berikut:

2.1.3.1 Enkapsulasi

Proses dimana setiap data dan fungsinya dikelompokkan kedalam suatu objek tersebut agar tidak dapat sembarang diakses atau diintervensi oleh program lain. Contoh enkapsulasi yaitu *Frame Ethernet.*

2.1.3.2 *Polymorphism*

Polymorphism adalah suatu proses untuk membuat sebuah fungsi yang sama dalam berbagai tipe objek yang berbeda. *Polymorphism* bertujuan untuk mengurangi objek yang ada dalam pemrograman sehingga *programmer* dapat menggunakan satu objek dengan metode yang berbeda.

2.1.3.3 *Inheritance*

Inheritance membolehkan suatu kelas untuk menurunkan atribut dan metode-metodenya untuk kelas yang lain agar dapat membuat sebuah kelas yang baru dan memiliki atribut dari kelas sebelumnya.

2.1.4 *Action Script 3.0*

Action Script 3.0 adalah suatu bahasa pemrograman yang dirancang untuk membuat animasi dan memudahkan *programmer* untuk menciptakan karya-karya istimewa. Animasi disini bisa dikatakan sebagai gambaran suatu *game* yang bertujuan memudahkan *programmer* dalam pembuatan *game* dengan tampilan animasi yang cukup baik.s

Pada *action script 3.0* menggunakan bahasa pemrograman berorientasi objek sehingga *programmer* dapat membangun lebih banyak fungsi pada suatu aplikasi yang kompleks.

2.1.5 Multimedia

Multimedia adalah suatu kombinasi antara suara, animasi, teks, dan video. Maksudnya adalah suatu aplikasi yang memiliki tingkat *interface* yang sangat tinggi dan menarik. *Multimedia* dapat berguna bagi banyak orang seperti dalam hal pendidikan, hiburan, dokumen pencitraan, dan jurnalisme. *Multimedia* memiliki beberapa elemen yaitu :

1. Teks

   Suatu informasi yang diberikan kepada *player* agar dapat memahami alur cerita. Namun terkadang teks ini dapat digunakan juga untuk memberi arahan kepada *player* agar dapat mengetahui langkah selanjutnya yang harus diambil. Akan tetapi, teks memiliki hal negatif yaitu jika alur cerita tidak menarik ataupun terlalu panjang *player* tidak akan tertarik dan meninggalkan *game* tersebut.

2. Gambar

   Gambar adalah suatu kumpulan dari piksel-piksel yang dijadikan satu dan membentuk sebuah gambar dengan beragam warna yang berbeda. Gambar berperan penting karena satu gambar dapat mewakili ribuan kata.

3. Suara

   Suara adalah suatu sarana untuk mendapatkan informasi dengan menggunakan telinga. Suara dapat digunakan untuk membuat efek, contoh ketika kita menyerang monster maka akan muncul efek suara, seperti suara yang menyerupai suara serangan pada pertarungan dunia nyata.

4. Animasi

Animasi adalah sekumpulan *frame* yang dijadikan satu dan dibuat dengan ketelitian yang sangat tinggi sehingga dapat menghasilkan karya yang luar biasa. Dengan animasi tampilan *game* dapat terlihat lebih baik sehingga lebih dapat menarik perhatian *player*.

2.1.6 Interaksi Manusia dan Komputer

Hubungan manusia dan komputer yang meliputi evaluasi, perancangan dan implementasi. Disini kita akan mempelajari bagaimana manusia berinteraksi dengan komputer dengan menggunakan Delapan Aturan Emas yaitu:

Delapan Aturan Emas *(Eight Golden Rules)*

Shneiderman (2011) mengemukakan "*Eight Golden Rules*" sebagai petunjuk dasar yang baik untuk merancang suatu *user interface*

1. Usahakan Untuk Konsisten

Urutan langkah yang sudah dilakukan harus mirip sesuai dengan tindakan yang telah dilakukan sebelumnya.

2. Memungkinkan *User* agar dapat menggunakan jalan singkat

User harus memiliki alternatif, untuk setiap tindakan yang telah dilakukanya, seperti memiliki tombol khusus agar memudahkan *user*.

3. Memberikan Umpan Balik yang Informatif

Sistem harus merespon atas tindakan yang telah dilakukan oleh *user*. Misalnya seperti memunculkan suara ketika tombol tersebut di kliki oleh *user*.

4. Mendesain Dialog untuk Menghasilkan Suatu Penutupan

Memberikan indikasi bahwa setiap tindakan yang dilakukan oleh *user* telah selesai dilakukan.

5. Penanganan Kesalahan yang Sederhana

Sebisa mungkin sistem dibuat agar *user* tidak melakukan kesalahan yang fatal. Jika terjadi kesalahan, maka sistem dapat mendeteksi kesalahan dengan cepat dan memberikan mekanisme yang sederhana dan mudah dipahami.

6. Mudah untuk Kembali ke Tindakan Sebelumnya

Jika *user* melakukan kesalahan. Maka system memilki alternatif atau pilihan yang dapat memungkinkan *user* agar dapat kembali ke tindakan sebelumnya.

7. menjadikan *user* sebagai pengendali

User ditempatkan sebagai pusat pengendali sistem. Karena *user* bebas melakukan tindakan apa saja dan sistem berjalan seperti keinginan *user.*

8. Mengurangi Beban Memori Jangka Pendek

Sistem harus mempermudah *user* agar dapat mengingat urutan tindakan yang dilakukan oleh *user*. Oleh sebab itu perlu adanya perancangan yang sederhana, agar dapat mempermudah user untuk mengingatnya.

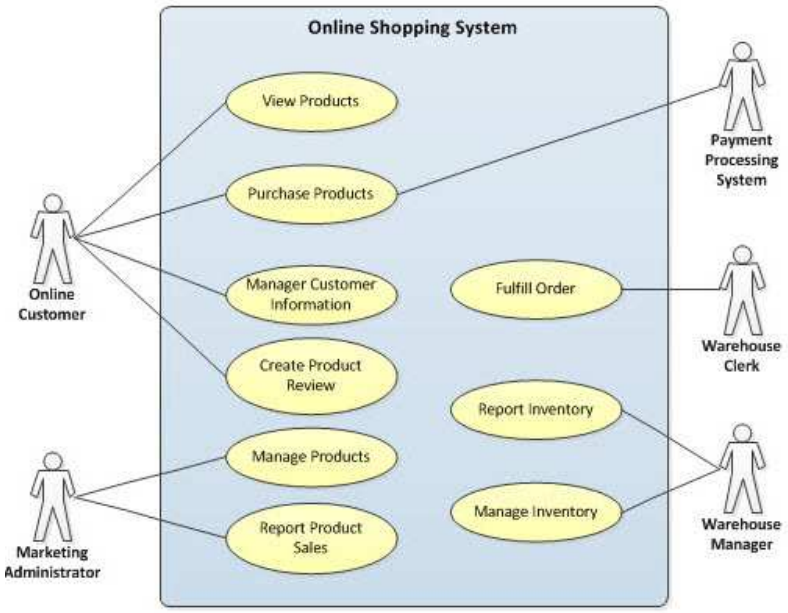
2.1.7 Unified Modeling Language

Unified Modeling Language adalah sebuah bahasa yang menjadi standar dalam merancang dan mendokumentasi sistem perangkat lunak. *unified modeling language* dapat merancang sebuah model dengan sistem sehingga semua jenis piranti lunak dapat membantu pendeskripsian dan desain sistem. *Unified modeling language* terdiri dari 5 kategori, yaitu:

2.1.7.1 Use-case Diagram

Use-case Diagram adalah suatu skema atau gambaran suatu sistem yang dirancang sebagai landasan dari suatu aplikasi. Maksudnya adalah melakukan perencanaan suatu sistem yang akan dibangun melalui sebuah gambar dan menunjukan langkah demi langkah dari perancangan tersebut.

Contoh :

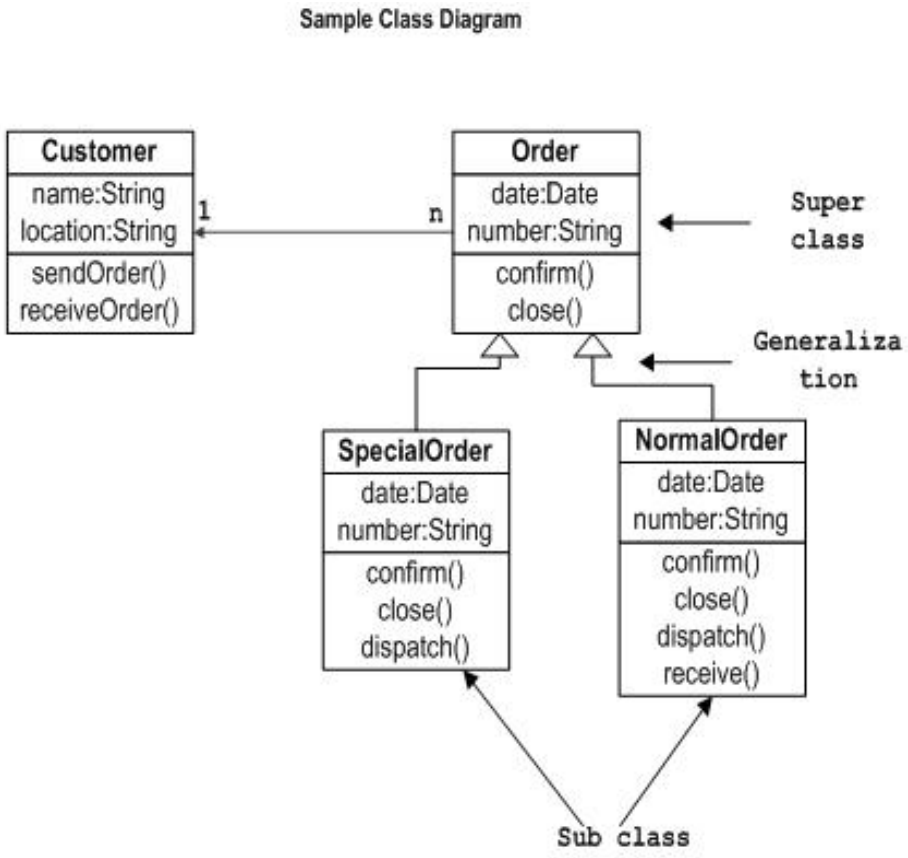


Gambar *2.3 Use-case Model* Diagram

2.1.7.2 Class Diagram

Mengambarkan struktur sebuah sistem dalam bentuk objek.menghubungkan *class* yang satu dengan yang lainnya dengan lebih rinci sehingga rancangan sistem terlihat lebih terencana.

Contoh :



Gambar *2.4 Class* Diagram

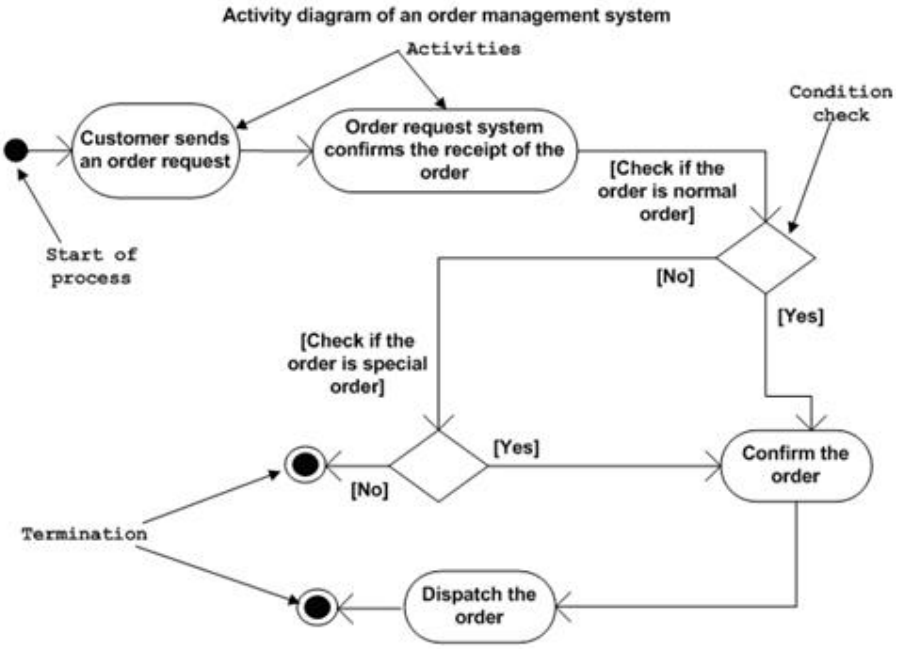
2.1.7.3 Interaction Diagram

Melakukan interaksi dengan menghubungkan antara suatu objek dengan objek yang lainnya dengan cara mengirimkan pesan. Contohnya adalah *Sequence* Diagram, pada diagram ini kita akan mengambarkan suatu diagram yang menunjukan cara kerja dari

sistem tersebut ketika pesan dikirimkan kepada sistem kemudian sistem akan merespon pesan tersebut.

2.1.7.4 Activity Diagram

Mengambarkan suatu urutan aktivitas secara berurutan dari sebuah *use-case* diagram. *Activity* diagram mengambarkan aliran proses yang terdapat dalam sistem mulai aktivitas dimulai start sampai aktivitas berhenti.



Gambar 2.5 *Activity* Diagram

2.1.7.5 *Implementation* Diagram

Implementation Diagram memodelkan struktur dari sistem informasi. Yang termasuk dari *implementation* diagram yaitu :

1. *Component* Diagram

Mengambarkan kode program yang dibagi-bagi menjadi modul sehingga dapat diprediksi

komponen apa yang dibutuhkan oleh sistem tersebut.

2. *Deployment* Diagram

Mengambarkan fisik dari *hardware* dan *software* didalam sistem. Disini kita membuat suatu gambaran sistem secara *run-time*.

2.2 Teori Khusus

2.2.1 Pengertian *Game*

Game merupakan suatu bentuk interaksi antara manusia dengan komputer yang menggunakan multimedia. *Game* adalah sarana penghibur untuk banyak orang terutama anak-anak. Akan tetapi, *game* dapat membahayakan anak atau orang-orang yang kecanduan untuk bermain.

2.2.2 Tipe Game

Tipe *game* adalah suatu ciri dari sebuah *game*. Setiap *game* memiliki ciri khas unik dan berbeda seperti pada penjelasan dibawah ini yang akan menjelaskan ciri-ciri dari sebuah *game*.

2.2.2.1 *Arcade*

Pada tipe *game* ini kecepatan adalah peranan penting untuk mencapai kemenangan. *Game* dengan tipe seperti ini biasanya menggunakan grafis dua dimensi.

2.2.2.2 *Puzzle*

Pada *game puzzle* ini lebih mengunakan kecerdasan *player* untuk menyelesaikan tantangan yang diberikan. Biasanya *game* dengan ciri khas ini menggunakan grafis dua dimensi.

2.2.2.3 *Racing*

Tipe *game* ini lebih cenderung melakukan balapan antara*player* maupunkomputer.

2.2.2.4 *Role Playing*

Pada *game role playing* yang dikenal dengan RPG ini biasanya adalah suatu *game* yang memiliki alur cerita dimana *player* akan menjalankan tantangan yang dikenal dengan nama "*QUEST"* dan mengikuti alur cerita.

Game dengan ciri khas ini memiliki alur cerita yang sangat luas sehingga untuk menyelesaikan *game* ini dibutuhkan waktu yang cukup lama. Didalam *game* ini *player* akan melawan berbagai macam monster yang telah disediakan. *Player* memiliki berbagai macam parameter sepeti *HP, MP, EXP,* dan *LEVEL*. Tipe *game* ini memiliki ciri khas yang sangat unik, sangat berbeda dengan tipe-tipe *game* lainnya.

2.2.2.5 *Sports*

Game sports adalah perpaduan dari berbagai macam *game* olahraga. Disini *player* akan berperan sebagai atlet dan akan mengikuti kontes-kontes olahraga sampai ketingkat yang sangat sulit.

2.2.2.6 *Strategy*

Tipe *game* ini lebih mengunakan kepintaran *player* untuk memenangkan *game*. Contoh:*Ice Cream Tycoon*, dimana pada *game* tersebut *player* diberi *goal* yang harus dicapai dalam kurun waktu tertentu.

2.2.3 Game Simulasi

Game simulasi adalah sebuah *game* yang menyerupai dunia nyata dengan tujuan yang berbeda-beda. Beberapa *game*

simulasi digunakan sebagai pelatihan yang bertujuan melatih seseorang untuk melakukan sesuatu tanpa membahayakan orang tersebut. Contoh pelatihan penerbangan, dengan adanya *game* simulasi untuk pelatihan penerbangan, seseorang dapat melatih dirinya untuk menerbangkan pesawat tanpa harus membahayakan dirinya.

2.2.4 Casual Game

Casual Game adalah suatu bentuk kesulitan *game* atau jenis dari *game* tersebut. Pada *Casual Game* biasanya memiliki kesulitan yang tidak terlalu sulit agar *player* dapat menikmati *game* tersebut, dengan tingkat kesulitan yang ringan dan tidak membutuhkan interaksi yang lebih menikmati *game* tersebut.

2.2.5 Game Design

Menurut Ernest Adams (2009), *Game Design* adalah suatu bentuk kegiatan mengimajinasikan suatu aplikasi. Dalam *Game Design* dibagi menjadi 3 kelompok yaitu:

2.2.5.1 Core Mechanics

Core Mechanics terdiri dari data dan algoritma yang mendefinisikan aturan utama didalam sebuah *game*. *Core mechanics* adalah bagian penting dari sebuah *game*, dapat dikatakan *Core Mechanics* adalah jantung atau jiwa dari sebuah *game*. Maka dari itu *Core Mechanics* mendapat perhatian yang lebih ketika kita mendesain sebuah *game*.

2.2.5.2 Storytelling dan Narasi

Dalam beberapa *game* memiliki cerita dimana cerita tersebut akan memberi informasi tentang isi dari *game* tersebut, cerita yang diberikan haruslah menarik sehingga dapat menarik perhatian banyak *player*. Tanpa adanya cerita didalam *game*,*game* tersebut akan kurang menarik,

maka dari itu suatu cerita dalam *game* berperan penting untuk suatu *game*.

Narasi adalah bagian dari cerita yang diceritakan oleh *game designer*.Biasanya narasi yang dibuat oleh *game designer* tidak terpengaruh oleh aksi *player* (bersifat linier).

2.2.5.3 Interaktivitas

Interaktivitas adalah suatu cara dimana *player* akan menentukan tingkat kenyamanan *game* tersebut, dengan mencakup beberapa hal seperti suara, grafik, *user interface* dan lain lain.

2.2.6 Game Engine

Game Engine adalah sebuah sistem perangkat lunak yang digunakan untuk mengembangkan sebuah *game*. dengan adanya *game engine* akan memudahkan setiap pembuat *game* untuk melakukan pengembangan terhadap sebuah *game*. *Game engine* memiliki *development tools* dengan tampilan visual yang langsung terintegrasi sehingga untuk melakukan pengembangan *game* menjadi lebih dipermudah dikarenakan *game engine*.

2.2.7 Stencyl

Stencyl adalah suatu aplikasi yang dibuat untuk membantu *programmer* membuat *game* 2D untuk perangkat *mobile*. Pada stencyl kita tidak hanya diberi kemudahan dalam membuat suatu *game* dengan cara*coding and drag and drop* saja akan tetapi tetap menggunakan logika.

Pada Stencyl kita sudah menghubungkan antara *actor*, *map* dan *coding* sudah dijadikan satu sehingga *programmer* tidak perlu membuat suatu pengabungan antara satu dengan yang lainnya.

2.2.8 Adobe Photoshop

Adobe Photoshop adalah aplikasi yang disediakan untuk melakukan *editing* pada gambar, dan pada *adobephotoshop* dapat melakukan *editing* melalui piksel-pikselnya sehingga dapat membuat gambar lebih menarik.Penciptaan gambar 3D pun dapat dilakukan dengan menggunakan *adobephotoshop* sehingga dalam satu aplikasi dapat melakukan banyak hal.

Adobe photoshop sangatlah berguna untuk mendesain gambar, poster, dan foto. Pengeditan dilakukan semata-mata untuk memperbaiki suatu gambar yang memiliki cacat atau ada bagian yang tidak ingin dimunculkan sehingga dilakukannya pengeditan melalui piksel-pikselnya.

2.3 Penelitian yang Terkait

Dalam jurnal "*Artificial Intelligence for Computer Games*" yang dibuat oleh Rhalibi, A,Wong, K.W. and Price, M (2009). *Artificial Intelligence* didalam *game* komputer meliputi *behavior* dan proses *decision making* dari lawan yang ada di *game* (yang dikenal sebagai NPC). Komputer dan *video games* generasi sekarang memiliki *testbed* yang menarik untuk penelitian *artificial intelligence* dan ide-ide barunya. Beberapa *game* menggabungkan kekayaan elemen dan lingkungan yang rumit dengan pengembangan yang sangat ahli dan stabil. *Game* komputer juga *multiagent*, membuat kerjasama, berkompetisi, dan memodelkan elemen-elemen menjadi sukses. Didalam *game* komersial seperti *gameaction*, *Role-Playing Games* (RPG*),* dan *game* strategi, perilaku dari NPC biasanya diimplementasi sebagai variasi dari *simple rule based system*. Dengan beberapa pengecualian, teknik *machine learning* sangat susah ketika diterapkan untuk *state of the art computer games*. Teknik *machine learning* memungkinkan NPC dengan kemampuannya untuk meningkatkan penampilannya dengan cara belajar dari kesalahan dan keberhasilan, untuk secara otomatis beradaptasi pada kekuatan dan kelemahan dari *player*, atau untuk belajar dari lawan mereka dengan meniru taktik mereka.

Dalam jurnal "*Investigate Naturalistic Decision-Making of Football Players in Virtual Environment: Influence of Viewpoints in Recognition"* yang dibuat oleh Yohann Cardin, Cyril Bossard, Cedric Buche dan Gilles Kermarrec (2013), menjelaskan bahwa *Decision Making* adalah suatu proses yang mendefinisikan suatu proses kognitif yang kompleks untuk menentukan aksi yang akan dilakukan dari pilihan aksi yang sudah ada dan dapat digunakan dalam situasi yang dinamis.

Dalam jurnal "*Solving the Physical Travelling Salesman Problem: Tree Search and Macro-Actions"* yang dibuat oleh Diego Perez et al. (2013). Menjelaskan bahwa *Travelling Salesman Problem* (TSP) adalah sebuah kombinasi untuk memecahkan masalah dalam pengambilan jarak terpendek antara setiap *node* atau *graph* dengan cara melintasi setiap *node* atau *graph* sebanyak satu kali, kemudian dapat ditentukan jarak terpendek yang bisa ditempuh.