BAB II
STUDI LITERATUR

2.1 Tinjauan Pustaka

Sampai saat ini ada beberapa penelitian sebelumnya yang mengimplementasikan *Finite State Machine* ke dalam game, maka dari itu penulis mencoba untuk mempelajari penelitian-penelitian sebelumnya untuk mengembangkan penelitian ini.

Penerapan *Finite State Machine* pada game biasanya banyak ditemukan untuk memberikan variasi respon pada NPC (*Non-Playable Character)* di dalam game, penggunaan *Finite State Machine* terhadap NPC ini juga turut diimplementasikan pada game berjudul "*The Relationship*" yang dikembangkan pada penelitian berjudul Penerapan Metode *Finite State Machine* Pada *Game "The Relationship*", di *game* ini sang penulis menggunakan *Finite State Machine* pada NPC semata-mata untuk memberikan kesan pada pemain sehingga seolah-olah NPC yang ada di game dapat terlihat cerdas [1]

Selain penelitian tersebut, penerapan *Finite State Machine* pada NPC juga digunakan pada penelitian berjudul "Pemetaan Perilaku *Non-Playable Character* Pada Permainan Berbasis *Role Playing Game* Menggunakan Metode *Finite State Machine*" pada penelitian kali ini pengimplementasian difokuskan kepada pengembangan game yang berbasis RPG (*Role Playing Game*) guna memberikan kesan *story* yang mendalam pada *game* [6], Selain menerapkan *Finite State Machine*, penelitian ini juga menggunakan *game engine* bernama RPG Maker XP sebagai *software* utama untuk membuat game pada penelitian ini [2]

Tidak hanya diterapkan pada NPC, pada penelitian berjudul "*Game* edukasi pengenalan budaya dan wisata kalimantan barat menggunakan *metode finite State machine* berbasis *android*" *Finite State Machine* sebagai cara untuk berpindah dari suatu kondisi ke kondisi yang lain untuk menawarkan pilihan

tingkah laku sesuai dengan aksi atau kejadian tersebut, pada penelitian ini juga membuktikan *Finite State Machine* dapat digunakan untuk *game* berbasis *android* [7].

2.2 Landasan Teori

Pada proses pengerjaan laporan ini, ada beberapa dasar teori yang digunakan sebagai acuan untuk melakukan penullisan. Antara lain:

2.2.1 Game

Menurut buku Pemrograman Animasi dan *Game* Profesional yang ditulis oleh Agustinus Nilwan dan diterbitkan oleh Elex Media Komputindo, *Game* adalah permainan komputer yang dibuat dengan menggunakan teknik dan metode animasi. Menurutnya pengetahuan tentang animasi sangatlah penting pada proses pembuatan game.[4]

Sedangkan menurut Clark C. Abt, *Game* merupakan kegiatan yang dimana pemain dapat membuat keputusan untuk mencapai tujuan game tersebut dengan dibatasi beberapa peraturan.[4]

Dari beberapa pengertian diatas dapat disimpulkan bahwa *Game* merupakan permainan yang menggunakan interaksi dengan pemain melalui gambar atau serangkaian animasi untuk menyelesaikan sebuah misi ataupun tujuan yang dibatasi dengan peraturan yang ada di dalam sistem[8].

Ada juga beberapa macam klasifikasi genre pada *game* berdasarkan dengan cara bermain yang bervariasi. Antara lain:

a. *Adventure,* adalah genre *game* dimana pemain harus berjalan untuk mengunjungi setiap tempat yang ada di dalam *game* [9].

b. *Action*, adalah genre *game* aksi dimana pemain biasanya mengandalkan reaksi cepat untuk bermain jenis *game* ini [10].

c. *Platformer*, adalah genre *game* dimana pemain harus menyelesaikan berbagai tingkatan level untuk menyelesaikan permainan[10].

d. *Puzzle*, adalah genre *game* dimana pemain membutuhkan kecerdasan untuk menyelesaikan teka-teki [10].

e. *Racing*, adalah genre *game* dimana pemain biasanya mengendarai kendaraan untuk berlomba menjadi yang tercepat [10].

f. *Role Playing Games,* adalah genre *game* dimana pemain menjadi suatu karakter tertentu dan bermain mengikuti cerita game tersebut [10]

g. *Strategy*, adalah *game* yang mengutamakan strategi untuk menyelesaikan permainan [10]

h. *Simulation*, adalah *game* yang mengutamakan sisi realistis dan dibuat semirip mungkin dengan dunia nyata [11].

Gambar 2.1 Contoh Game 2D Action

Pada penelitian ini penulis memilih genre game *Action* untuk dikembangkan dan kemudian diimplementasi kan *Finite State Machine*.

2.2.2 Game Engine Unity

Unity adalah salah satu dari banyak game engine yang biasa digunakan sebagai *tool* atau *software* untuk membuat *game*,[12] Unity dibuat oleh Unity Technologies Inc. Unity dapat digunakan untuk membuat *game* berbasis PC maupun *mobile*, selain dukungan platform yang cukup banyak, Unity juga mendukung tiga bahasa pemrograman antara lain Javascript, C# dan Boo. Fitur

Scripting di Unity sangatlah mudah untuk digunakan terlebih dengan tersedianya editor Visual Studio[13]

Selain fitur Scripting, *Game engine* yang satu ini juga memiliki berbagai fitur lain. Yaitu:

a. *Asset Tracking* dan *Asset Store*, di dalam Unity terdapat pengelolan *asset* yang dapat mempermudah proses pengembangan, selain itu terdapat juga *Asset Store* yang menyediakan berbagai macam resource dari mulai *materials, textures, models* bahkan *sound effects* yang bisa digunakan pada proses pengembangan [14].

b. *Physics*, merupakan fitur yang dapat menambahkan simulasi *pysics* untuk menambah kesan realistis [14].

c. *Platforms,* merupakan fitur yang dipunyai *Unity* untuk Mendukung proses pengembangan *Multiplatform* [15]

d. *Rendering*, merupakan fitur yang disediakan Unity untuk mengimplementasikan Asset ke dalam *game project* dan diatur melalui GUI (*Graphical User Interface*) Unity, Format desain yang banyak digunakan adalah Blender, Adobe Photoshop dll. Kemampuan lain dari fitur *rendering* juga dapat melakukan *Screen Space Ambient Occlusion* (SSAO), *dynamic shadow, reflection, parallax* dan *render-to-texture* [16].

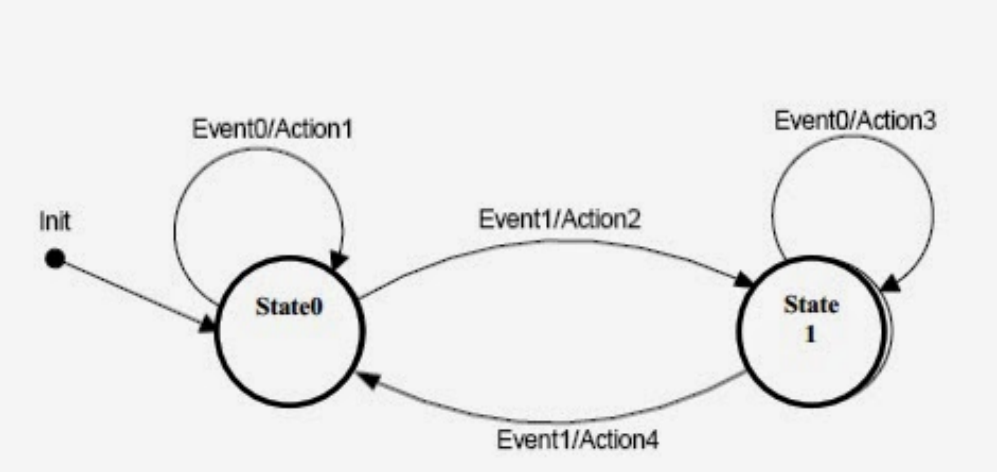
Gambar 2.2 Logo Game Engine Unity

2.2.3 Finite State Machine

Sebuah finite state machine adalah model komputasi yang dapat digunakan untuk mensimulasikan urutan logis dan untuk mewakili urutan dan kondisi eksekusi [12]. Sebuah mesin keadaan terbatas terdiri dari serangkaian keadaan yang menentukan kapan harus membuat keputusan, serta transisi setiap keadaan ke keadaan berikutnya ketika kondisi sebelumnya terpenuhi [17].

Prinsip pengoperasian finite state machine dapat dibagi menjadi tiga bagian: state, event, dan action [18]. Pada titik tertentu sistem menjadi aktif, dan ketika input atau peristiwa diterima, sistem beralih ke keadaan lain dan proses switching melibatkan tindakan yang diambil oleh sistem. Diimplementasikan dalam menanggapi peristiwa baru-baru ini. [6]. Proses transfer umumnya juga mencakup proses yang relatif sederhana atau kompleks [18].

Mesin keadaan terbatas juga cocok untuk digunakan dalam proyek perangkat lunak kontrol waktu nyata yang reaktif. Ini karena kemampuan mesin keadaan terbatas untuk membagi aplikasi besar menjadi sejumlah kecil item keadaan dapat sangat membantu proses desain perangkat lunak. Selain itu dapat digunakan untuk mendesain game dan aplikasi lainnya [18].



Gambar 2.3 Contoh Diagram Finite State Machine