



Hak cipta dan penggunaan kembali:

Lisensi ini mengizinkan setiap orang untuk menggubah, memperbaiki, dan membuat ciptaan turunan bukan untuk kepentingan komersial, selama anda mencantumkan nama penulis dan melisensikan ciptaan turunan dengan syarat yang serupa dengan ciptaan asli.

Copyright and reuse:

This license lets you remix, tweak, and build upon work non-commercially, as long as you credit the origin creator and license it on your new creations under the identical terms.

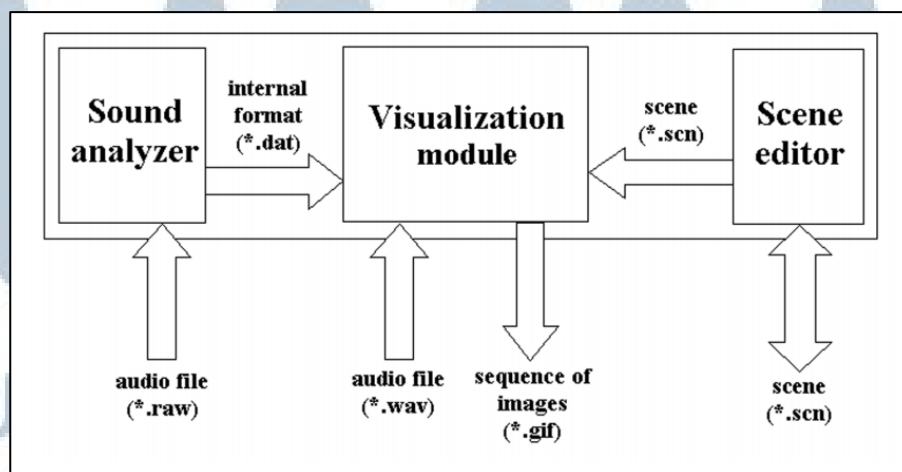
BAB II

LANDASAN TEORI

2.1 Visualisasi Audio

Aspek yang harus dilihat dalam membuat sebuah visualisasi audio yaitu *volume, mood, melody, instrument, tempo*, efek visual, dan lain-lain. Setelah itu, semua aspek tersebut dipadukan untuk membuat sebuah visualisasi yang harmoni (Kubelka, 2000). Maka dari itu, diperlukan cara untuk memadukan semuanya itu agar bisa berjalan bersinergi.

Untuk memadukannya, ada sebuah skema yang dapat membantu pembuatan visualisasi audio. Seperti yang ditunjukkan pada Gambar 2.1, visualisasi audio memiliki sebuah alur atau skema dan memiliki komponen utama yaitu *sound analyzer, visualization module* dan *scene editor* (Kubelka, 2000).



Gambar 2.1 Skema Basis Dari Video Game Visualisasi Audio (Kubelka, 2000)

2.1.1 Sound Analyzer

File audio yang masuk ke dalam *sound analyzer*, dianalisa dan dibagi menjadi berbagai bagian. Biasanya ada 5, yaitu sebagai berikut (Kubelka, 2000).

- *Volume*, yaitu besar kecilnya suara sebuah audio.
- *Balance*, yaitu posisi audio stereo (lebih condong kiri / kanan / di tengah).
- *Dominant (solo) instrument position (melody)*.
- *Composition rate* (tempo).
- *Mood*, yaitu estimasi dari *feel* musik.

Frekuensi suara yang dapat telinga manusia tangkap adalah berkisar dari 20Hz-20.000Hz (Rosen, 2011). Kisaran frekuensi suara ini dapat dijadikan berbagai golongan seperti yang ditunjukkan oleh Gambar 2.2.

Frequency (Hz)	Octave	Description
16 to 32	1st	The lower human threshold of hearing, and the lowest pedal notes of a pipe organ.
32 to 512	2nd to 5th	Rhythm frequencies, where the lower and upper bass notes lie.
512 to 2048	6th to 7th	Defines human speech intelligibility, gives a horn-like or tinny quality to sound.
2048 to 8192	8th to 9th	Gives presence to speech, where labial and fricative sounds lie.
8192 to 16384	10th	Brilliance, the sounds of bells and the ringing of cymbals and sibilance in speech.
16384 to 32768	11th	Beyond Brilliance, nebulous sounds approaching and just passing the upper human threshold of hearing

Gambar 2.2 Golongan Frekuensi Suara (Kubelka, 2000)

Data musik atau *spectrum data* (tipe data *float*) dari musik yang sedang diputar perlu diambil. *Spectrum data* dapat diambil oleh fungsi `GetSpectrumData()` dalam IDE (*Integrated Development Environment*) Unity. Di dalam fungsi ini diperlukan beberapa *parameter* yaitu, variabel tempat menampung nilai, *channel*-nya (*boolean*, *output speaker* kiri atau kanan), dan Fast Fourier Window (FFTWWindow, adalah fungsi matematika yang dapat mengambil *spectrum data* tadi). Fungsi ini mengembalikan nilai berupa hasil perubahan *spectrum data* menjadi data yang bisa dipakai dalam *game* (Unity Technologies, 2017). *Spectrum data* yang sudah didapatkan dapat juga ditampung ke dalam sebuah *array* jika objek yang nanti akan bergerak itu banyak.

2.1.2 Scene Editor

Pada bagian ini, objek dalam *scene* harus di-*define* terlebih dahulu. Objek itu bisa berupa *basic geometric object* (*sphere*, *cube*, dan lain-lain) atau partikel sistem. Objek-objek ini terbentuk dari pergerakan data yang berganti-ganti dari parameter *sound analyzer*. Setelah itu, *scene* diproses di *visualization module* untuk ditampilkan.

Objek visualisasi audio berupa *basic geometric object* tidak dapat dibuat kompleks atau abstrak. Dalam pembuatan visualisasi audio, perlu ada objek yang memiliki bentuk abstrak, namun tetap bisa dapat dikontrol. Objek juga harus dapat mengekspresikan *feel* dari sebuah music dengan baik. Dan yang dapat menjawab semua hal itu adalah *particle system* (Kubelka, 2000).

2.1.3 Visualization Module

Tugas dari *visualization module* adalah memproses *scene data* dan *sound data*. Tahap ini mengatur *rendering speed* tiap fase animasi yang terjadi. Pada dasarnya, performa bergantung pada kehandalan *equipment* pengguna. Kecepatan optimal untuk me-render satu frame animasi adalah 1/25 sec (25 fps). Dalam *real-time*, pasti ada beberapa *scene* yang sulit di-render dalam kecepatan tersebut, khususnya saat memasuki *scene* yang kompleks.

Maka dari itu, *visualization module* bekerja dalam 2 mode yaitu sebagai berikut (Kubelka, 2000).

1) Preview mode

Mode ini untuk membantu *user* melihat *preview* animasi sehingga bisa diestimasi *design* animasi untuk selanjutnya.

2) Full-render mode

Dipakai untuk me-render hasil akhir sebuah animasi.

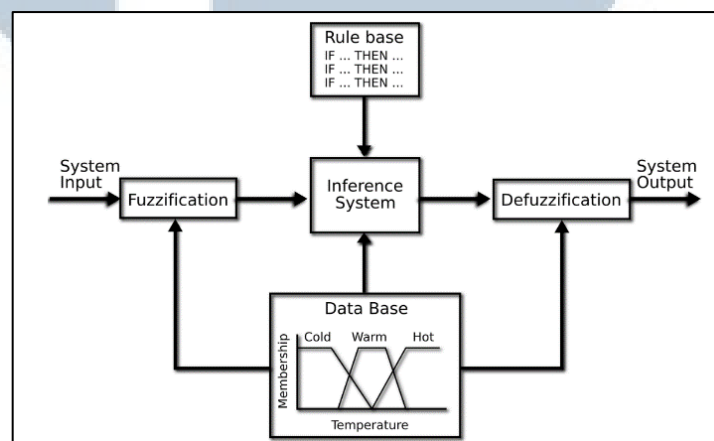
2.2 Algoritma Fuzzy Logic

Algoritma *fuzzy logic* adalah peningkatan dari logika Boolean yang berhadapan dengan konsep kebenaran sebagian. Saat logika klasik menyatakan bahwa segala hal dapat diekspresikan dalam istilah biner (0 atau 1, hitam atau putih, ya atau tidak), *fuzzy logic* menggantikan kebenaran boolean dengan tingkat kebenaran. *Fuzzy logic* memungkinkan nilai keanggotaan antara 0 dan 1, tingkat keabuan dan juga hitam dan putih, dan dalam bentuk linguistik, konsep tidak pasti seperti "sedikit", "lumayan", dan "sangat". Logika ini berhubungan dengan *fuzzy set* dan teori

kemungkinan. Logika fuzzy diperkenalkan oleh Dr. Lotfi Zadeh dari Universitas California, Berkeley pada 1965 (Novák, 1999).

Fuzzy logic umumnya diterapkan pada masalah masalah yang mengandung unsur ketidakpastian (*uncertainty*), ketidaktepatan (*imprecise*), *noisy*, dan sebagainya. *Fuzzy logic* menjembatani bahasa mesin yang presisi dengan bahasa manusia yang menekankan pada makna atau arti (*significance*). (Novák, 1999)

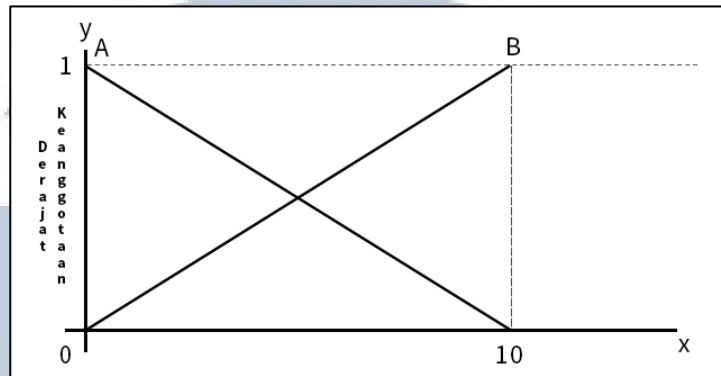
Gambar 2.3 menunjukkan, ada tiga proses utama jika ingin mengimplementasikan *fuzzy logic* pada suatu perangkat, yaitu *fuzzification*, *rule evaluation*, dan *defuzzification* (Zadeh, 1996).



Gambar 2.3 Gambar Arsitektur Fuzzy Logic (Zadeh, 1996)

2.2.1 Fuzzification

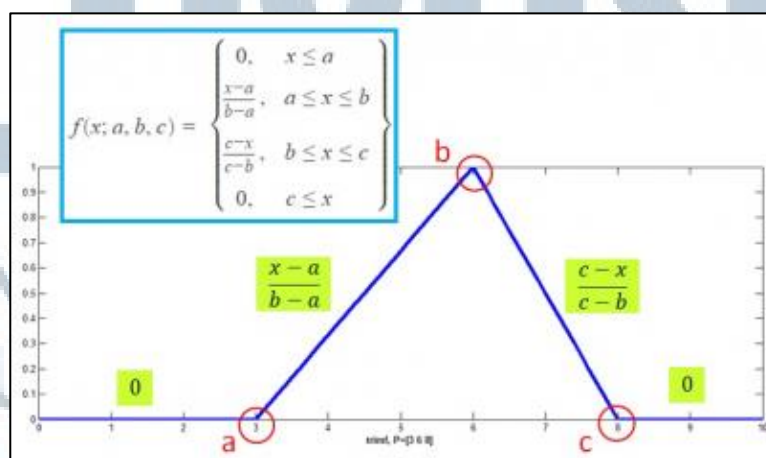
Merupakan suatu proses untuk mengubah suatu masukan dari bentuk tegas (*crisp*) menjadi *fuzzy* yang biasanya disajikan dalam bentuk himpunan-himpunan *fuzzy* dengan suatu fungsi keanggotaannya masing-masing (Zadeh, 1996). Contoh gambar fungsi keanggotaan sebuah himpunan *fuzzy* dapat dilihat pada Gambar 2.4.



Gambar 2.4 Contoh Gambar Fungsi Keanggotaan (Zadeh, 1996)

Dari Gambar 2.4, dapat diketahui bahwa ada sebuah fungsi keanggotaan yang memiliki dua himpunan yaitu A dan B. Sumbu x merepresentasikan masukan dari algoritma *fuzzy* dan sumbu y merepresentasikan derajat keanggotaannya (nilai *fuzzy*). Pada contoh di atas masukan fuzzy memiliki rentang nilai dari 0 sampai 10 dan derajat keanggotaan memiliki rentang nilai dari 0 sampai 1.

Untuk mendapatkan nilai *fuzzy* digunakan rumus persamaan garis lurus yang melalui titik potong sebuah himpunan *fuzzy*. Rumus ini dapat dilihat pada Gambar 2.5 (Hazewinkel, 2001).



Gambar 2.5 Rumus Persamaan Garis Lurus (Kubelka, 2000)

Setelah memperoleh derajat keanggotaan untuk tiap-tiap input, langkah selanjutnya adalah menerapkan operator-operator logika atau biasanya sering disebut aturan-aturan. Aturan-aturan ini berupa logika jika-maka yang sudah ditetapkan sebelumnya. Salah satu contoh aturan logika yang cocok untuk Gambar 2.4 adalah jika input X, maka akan melakukan proses B. Nilai X merupakan sebuah masukan dan proses B ini adalah *output* yang diharapkan.

2.2.2 Inference System (Rule Evaluation)

Proses ini mengevaluasi aturan yang sudah ditentukan sebelumnya sebagai acuan untuk menjelaskan hubungan antara variabel-variabel masukan dan keluaran. Variabel yang diproses dan yang dihasilkan berbentuk *fuzzy*. Untuk menjelaskan hubungan antara masukan dan keluaran ini, dapat digunakan metode Tsukamoto (Zadeh, 1996).

Pada metode Tsukamoto, implikasi tiap aturan berbentuk implikasi “Sebab-Akibat” atau “Input-Output” (Zadeh, 1996). Sehingga ada dua cara menentukan hubungan antar variabel yaitu sebagai berikut.

1) Konjungsi *fuzzy*

$$\begin{aligned}\alpha &= \mu A \wedge B = \mu A(x) \cap \mu B(y) \\ &= \min(\mu A(x), \mu B(y)) \quad \dots (2.2)\end{aligned}$$

2) Disjungsi *fuzzy*

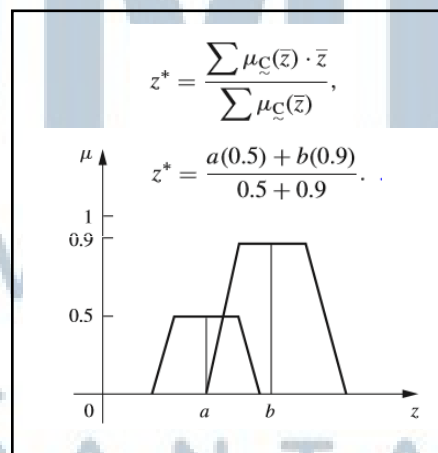
$$\begin{aligned}\alpha &= \mu A \vee B = \mu A(x) \cup \mu B(y) \\ \alpha &= \max(\mu A(x), \mu B(y)) \quad \dots (2.1)\end{aligned}$$

Penggunaan logika ‘atau’, atau ‘*or*’ mengindikasikan pemakaian cara disjungsi dalam perhitungan, yaitu mencari nilai yang paling besar. Sedangkan, Penggunaan logika ‘dan, atau ‘*and*’ mengindikasikan pemakaian cara konjungsi dalam perhitungan, yaitu mencari nilai yang paling kecil.

Evaluasi aturan hubungan antar variabel menghasilkan derajat keanggotaan dari *output* yang diharapkan. Derajat keanggotaan ini selanjutnya dipakai untuk menentukan nilai-nilai pada fungsi keanggotaan *output*. Cara menentukan nilai ini dapat digunakan rumus persamaan garis lurus yang dapat dilihat pada Gambar 2.4.

2.2.3 Defuzzification

Merupakan proses pengubahan variabel berbentuk *fuzzy* tersebut menjadi data-data pasti (*crisp*) agar data tersebut dapat dipakai (Zadeh, 1996). Salah satu cara yang dapat digunakan adalah menggunakan metode *weighted average*. Metode ini paling sering digunakan pada program atau aplikasi yang menggunakan *fuzzy logic* karena metode ini efisien jika dikomputasikan. Contoh penggunaan metode *weighted average* dapat dilihat pada Gambar 2.6.



Gambar 2.6 Metode Defuzzifikasi *Weighted Average*

2.3 Keinteraktifan Suara dalam Video Game

Suara dalam media interaktif seperti *video game* adalah *multimodal*, yang artinya adalah interaksi yang menggunakan lebih dari satu *sensor modality* dan biasanya terdiri dari 3 (*vision, audition, and haptics-action, image, and sound*). (Collins, 2013)

Synchresis adalah istilah yang berarti ketika visual dan suara saling bersistensis dan sinkron (berasal dari kata *synchronism* dan *synthesis*) (Chion, 1994). Konsep ini ada karena keahlian otak yang berupaya menggabungkan *sensory input* yang berbeda sehingga menyebabkan sebuah ilusi. Contoh sederhananya adalah ketika melihat orang yang sedang berjalan, diharapkan ada suara orang berjalan.

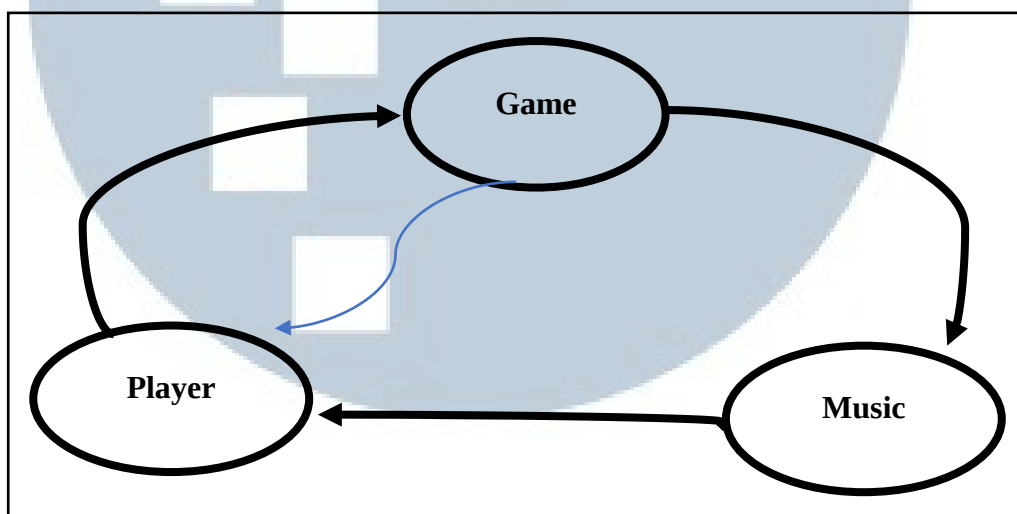
Dengan mempertimbangkan efek *synchresis*, patut diperhatikan juga dalam pemakaian *sound effect* dalam membuat interaktif dalam sebuah video game. Tergantung suasana yang terjadi, *output* dari *synchresis* dapat berbeda-beda seperti, saling mendukung (*agonistic* atau *congruent*, seperti contoh di atas), saling menolak (*antagonistic* atau *congruent*), dan tidak ada efek sama sekali (*neutral*) (Chion, 1994).

Interactive music dalam konteks video game adalah kondisi ketika *video game* sadar terhadap musik yang sedang diputar sehingga *game* dapat merespon sesuai dengan suara musik tersebut (Stevens & Raybould, 2011: 237).

Ada berbagai jenis untuk mendeksripsikan tipe musik (atau suara) dalam sebuah game, yaitu sebagai berikut (Stevens & Raybould, 2011: 237).

2.3.1 Reactive Music System = Adaptive or Dynamic Music

Gambar 2.7 menunjukkan bahwa musik disini menanggapi aksi dari *player* sebagai penengah dari *game state* atau yang lainnya tergantung *game* itu sendiri. Musik hanya berperan sebagai *recipient* dari sebuah *game engine*, berjalan secara *linear* menunggu perintah dari sistem *game*.

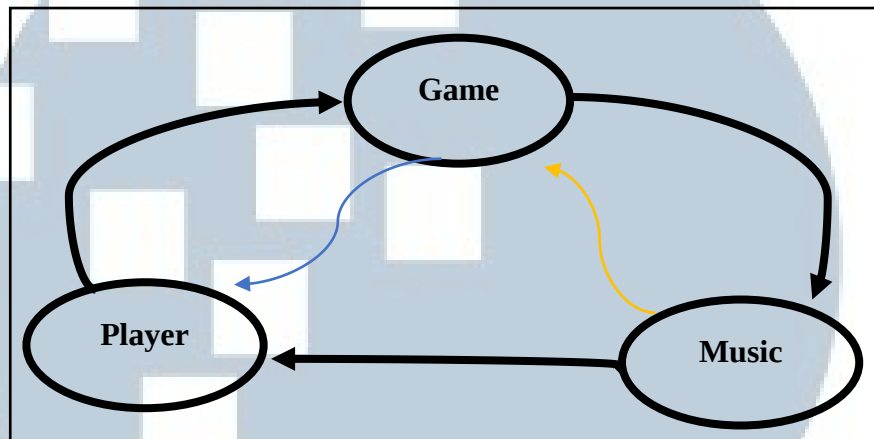


Gambar 2.7 *Reactive Music System* (Stevens & Raybould, 2011: 237)

2.3.2 Interactive Music Systems

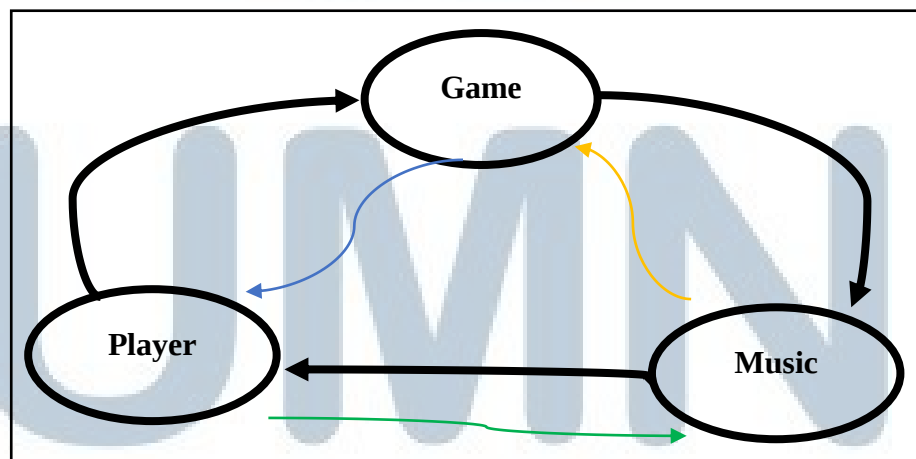
Untuk benar-benar membuat musik menjadi interaktif, dibutuhkan *feedback cycle* diantara *game engine* dan *music state*. Seperti yang ditunjukkan Gambar 2.8, bukan hanya menerima perintah dari *game* dan meresponnya, melainkan *game engine* harus menerima *input/feedback* dari musik itu sendiri. Input yang bisa diberikan ke *game engine* misalnya posisi musik sekarang (sedang di menit ke berapa), atau waktu sisa untuk transisi musik selanjutnya. Dengan begini, *game*

state bisa menentukan kapan harus memanggil musik secara langsung atau menunggu masukan dari *music data*.



Gambar 2.8 *Interactive Music System* (Stevens & Raybould, 2011: 237)

2.3.3 Performative Systems



Gambar 2.9 *Performative Music System* (Stevens & Raybould, 2011: 238)

Gambar 2.9 menunjukkan bahwa musik interaktif dapat memungkinkan *player* punya kendali langsung terhadap musik itu sendiri. Contohnya seperti *music-based games* atau *rhythm-action genre*.

2.4 Game Design

Ada dua elemen yang perlu diketahui dalam mendesain sebuah *game* yaitu *formal elements* dan *dramatic elements* (Chritopher, Steven, & Tracy, 2008). *Formal elements* adalah elemen-elemen yang menyusun struktur *game*. Tanpa *formal elements*, sesuatu tidak dapat disebut sebagai sebuah *game*. Terdapat delapan *formal elements* seperti dijabarkan Fullerton, antara lain sebagai berikut.

- 1) *Players*, yaitu pemain *game*. Elemen ini membahas tentang berapa jumlah pemain, peran dari tiap pemain, dan pola interaksi pemain.
- 2) *Objective*, adalah suatu tujuan yang diberikan pada pemain untuk pemain perjuangkan atau usahakan. Mendefenisikan apa yang harus pemain capai dalam permainan ini.
- 3) *Procedures*, yaitu metode dan aksi yang dapat dilakukan pemain untuk mencapai *objective* dalam *game*.
- 4) *Rules*, yaitu aturan yang membatasi aksi pemain dalam ruang lingkup *video game*. *Rules* disini juga bisa diartikan dengan mendefenisikan objek *video game*. Terdapat beberapa *rules* yang umum antar *game*, diantaranya adalah *rules* yang mendefinisikan objek dan konsep, *rules* yang membatasi tindakan pemain, dan *rules* yang menentukan akibat dari tindakan tertentu.
- 5) *Resources*, yaitu sumber daya berupa aset-aset yang dapat digunakan untuk menyelesaikan tujuan tertentu dalam *game*.
- 6) *Conflict*, yaitu rintangan atau konflik yang muncul akibat pemain yang berusaha mencapai *objective game* dalam keadaan *rules* dan *procedures* yang menghalangi pemain untuk mencapai tujuan tersebut secara langsung.

- 7) *Boundaries*, yaitu batasan antara *game* dan segala sesuatu yang tidak termasuk didalam *game*.
- 8) *Outcome*, yaitu hasil akhir dari *game*. *Outcome* dari *game* harus bersifat tidak pasti, untuk menarik minat para pemain.

Sedangkan, *dramatic elements* adalah elemen-elemen yang membuat pemain merasakan pengalaman bermain secara emosional sehingga pengalaman itu melekat pada pemain. *Dramatic elements* memberi konteks pada *gameplay* dan dengan mengintegrasikannya dengan *formal elements*, pengalaman bermain menjadi berarti bagi pemain. Tidak seperti *formal elements*, *dramatic elements* tidak wajib disertakan semuanya dalam *game*. Terdapat tujuh *dramatic elements*, antara lain sebagai berikut.

- 1) *Challenge*, yaitu tugas atau rintangan dalam *game* yang terasa memuaskan jika diselesaikan. Untuk menyelesaikannya juga butuh jumlah usaha yang tepat untuk menciptakan sebuah rasa pencapaian atau prestasi dan kegembiraan.
- 2) *Play*, yaitu kebebasan bergerak atau kendali saat berada dalam *game* yang sudah terlimitasi. Terlimitasi disini disebabkan karena *rules* dan *procedures* dari *formal elements*. Kebebasan yang dimaksud adalah kebebasan pemain untuk bertindak selama masih mematuhi *rules*.
- 3) *Premise*, yaitu mengimplementasikan unsur drama dalam *game* agar pemain dapat mentoleransi keadaan yang dibuat dalam *formal elements*. Tanpa *premise* yang dramatis, *game* akan terlalu abstrak bagi pemain untuk secara emosional terikat atau peduli ke dalam *game*.

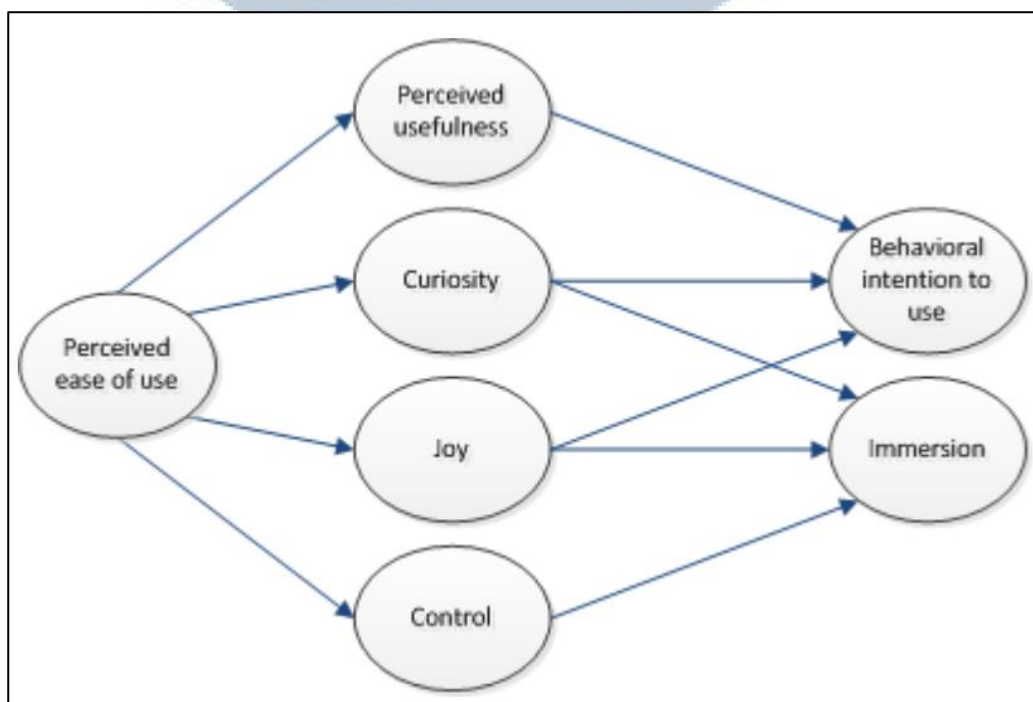
- 4) *Character*, yaitu agen-agen yang melakukan tindakannya sesuai drama yang diceritakan.
- 5) *Story*, yaitu cerita dalam permainan. Dalam hal menceritakan sesuatu, cerita pada *game* berbeda terhadap cerita pada film, televisi, dan pertunjukkan, karena cerita pada *game* ditentukan dan diselesaikan oleh pemain sendiri.
- 6) *World building*, yaitu membuat cerita atau kesan yang terbentuk dari sebuah dunia fiksi. *World building* mengandung budaya, sejarah, budaya, bahasa, mitologi, pemerintahan, politik, ekonomi dan lain-lain. Sehingga unsur ini perlu desain dan penelitian yang mendalam sebelum membuatnya.
- 7) *The dramatic arc*, yaitu komposisi ketegangan dramatis dalam cerita seiring berjalannya *game*.

Ada banyak kemungkinan kombinasi yang dapat tercipta dari elemen-elemen ini sehingga kombinasi kedua elemen ini dapat menciptakan berbagai pengalaman baru bagi pemain. Dengan memiliki pemahaman tentang bagaimana elemen-elemen ini bekerja sama dan bertindak terhadap *game* yang dibuat, dapat dipastikan permainan yang dibuat akan memberikan sebuah inovasi baru dalam industri *game* (Christopher, Steven, & Tracy, 2008).

Game yang telah didesain didokumentasikan ke dalam Game Design Document (GDD). GDD adalah dokumen tertulis yang selalu diperbarui dan sangat deskriptif mendeskripsikan *design* dari video game agar tim developer mempunyai sebuah acuan dalam membuat sebuah video game (Bethke, 2003).

2.5 Hedonic-Motivation System Adoption Model (HMSAM)

HMSAM adalah salah satu alternatif model yang bisa dipakai dalam *technology acceptance model* (TAM) (Lowry dkk., 2013). TAM merupakan salah satu model yang dibangun untuk menganalisis dan memahami faktor-faktor yang mempengaruhi diterimanya penggunaan teknologi komputer. Namun, TAM tidak ideal atau cocok dalam menjelaskan pemakaian dari murni *intrinsic* atau sistem *hedonic* (contohnya *online games*, musik, belajar tentang kepuasan, dan lain lain). Maka, dibuatlah sebuah sistem model alternatif, yang bernama HMSAM (Lowry dkk., 2013). Untuk lebih jelas, pada Gambar 2.10 ditunjukkan struktur model HMSAM.



Gambar 2.10 Struktur Model HMSAM (Lowry dkk., 2013)

Skala pengukuran pada penelitian HMSAM dapat dilihat pada Gambar 2.11.

Appendix 1. Measurement Scales

Table A1.1 Measurement Scales Table		
Construct	Items	Notes
Joy	JOY1. I found playing the game to be enjoyable.	Built on the original three-item scale from (Venkatesh, 2000) by modifying to gaming context and added three items based on hedonic enjoyment concepts from the following literature: (Agarwal & Karahanna, 2000; Igbaria et al., 1995; Raney et al., 2003; van der Heijden, 2004).
	JOY2. I had fun using the game.	
	*JOY3. Using the game was boring.	
	*JOY4. The game really annoyed me	
	JOY5. The game experience was pleasurable.	
	*JOY6. The game left me unsatisfied.	
Control	CTL1. I had a lot of control.	Modified original scale from (Liu, 2003) to be more general so that it did not focus solely on Web sites. Also, added items on control from (Agarwal & Karahanna, 2000).
	CTL2. I could choose freely what I wanted to see or do.	
	*CTL3. I had little control over what I could do.	
	CTL4. I was in control.	
	*CTL5. I had no control over my interaction.	
	CTL6. I was allowed to control my interaction.	
Focused Immersion	FI1. I was able to block out most other distractions.	Modified original scale from (Agarwal & Karahanna, 2000) to a gaming context; third item was changed from task to game.
	FI2. I was absorbed in what I was doing.	
	FI3. I was immersed in the game.	
	*FI4. I was distracted by other attentions very easily.	
	(d1)(d2) FI5. My attention was not diverted very easily.	
Temporal Dissociation	TD1. Time appeared to go by very quickly using the game.	Modified original scale from (Agarwal & Karahanna, 2000) to a gaming context; last two original items were not included because they did not fit our context.
	TD2. I lost track of time when I was playing the game.	
	TD3. Time "flew" when I played the game.	
Curiosity	CUR1. This experience excited my curiosity.	Modified original scale from (Agarwal & Karahanna, 2000) to a gaming context.
	CUR2. This experience made me curious.	
	CUR3. This experience aroused my imagination.	
Perceived Ease-of-Use	PEOU1. My interaction with the game was clear and understandable.	Combined four-item scale from (Agarwal & Karahanna, 2000) and four-item scale from (Venkatesh, 2000) and modified to a gaming context.
	PEOU2. Interacting with the game did not require a lot of my mental effort.	
	PEOU3. I found the game to be trouble free.	
	PEOU4. I found it easy to get the game to do what I want it to do.	
	(d1) PEOU5. Learning to operate the game was easy for me.	
	PEOU6. It was simple to do what I wanted with the game.	
	PEOU7. It was be easy for me to become skillful at using the game.	
Perceived Usefulness	PU1. The game decreased my stress.	Modified original utilitarian scale from (Venkatesh, 2000) to a hedonic context. Items chosen from pilot test items that showed the strongest benefits of fun.
	PU2. The game helped me better pass time.	
	PU3. The game provided a useful escape.	
	PU4. The game helped me think more clearly.	
	PU5. The game helped me feel rejuvenated.	
Behavioral Intention to Use	BIU1. I would plan on using it in the future.	Modified original scale from (Agarwal & Karahanna, 2000) to a gaming context
	BIU2. I would intend to continue using it in the future.	
	BIU3. I expect my use of it to continue in the future.	

Notes on measures:

- *=reverse scaled; d1=dropped for Study 1
- All scales were reflective and used a Likert-like seven-point scale anchored on "Strongly Disagree" to "Strongly Agree."
- The post-experiment measures were virtually the same for both Study 1 and Study 2. However, since Study 1 required the participants to imagine the games rather than actually play them, all prompts were changed to be conditional (e.g., "I would" instead of "I was"). To get participants in Study 1 to imagine the effects of the games more vividly, we asked them to imagine playing their assigned game for two hours.

Gambar 2.11 Skala Pengukuran Model HMSAM (Lowry dkk., 2013)

Dalam penelitian Lowry (2013), diajak beberapa orang untuk melakukan eksperimen lalu peserta eksperimen mengisi sebuah survei. Seperti yang dapat terdapat pada Gambar 2.11, skala pengukuran dalam survey ini mengikuti model HMSAM. Skala pengukuran ini dipakai dalam penelitian *game* Beat Defender sebagai dasar dalam pembuatan survei.

2.6 Skala Likert

Skala Likert merupakan metode pengukur tanggapan positif ataupun negatif terhadap suatu pernyataan. Skala pilihan dapat berupa genap atau ganjil. Perbedaannya adalah jika skala pilihan ganjil, kuesioner skala Likert terdapat pilihan netral di tengah. Sedangkan jika skala pilihan genap, kuesioner skala Likert memaksa orang memilih salah satu kutub karena pilihan netral tak tersedia. Untuk ukuran sampel, minimal diambil paling sedikit 30, 50, 75, 100 atau kelipatannya (Riduwan, 2009).

Dalam membuat skala Likert, ada beberapa langkah prosedur yang harus dilakukan peneliti, antara lain (Moh, 2005 ; Sugiyono 2008) sebagai berikut.

1. Mengumpulkan pernyataan yang memiliki relevansi dengan masalah yang sedang diteliti dan terdiri dari pernyataan yang cukup jelas atau tidak disukai.
2. Kemudian pernyataan itu dicoba kepada sekelompok responden yang cukup representatif dari populasi yang ingin diteliti.
3. Responden di atas diminta untuk mengecek tiap pernyataan, apakah ia menyenangkan (+) atau tidak menyukainya (-). Respons tersebut dikumpulkan dan jawaban yang memberikan indikasi menyenangkan diberi skor tertinggi.

Tidak ada masalah untuk memberikan angka 5 untuk yang tertinggi dan skor 1 untuk yang terendah atau sebaliknya. Yang penting adalah konsistensi dari arah sikap yang diperlihatkan. Demikian juga apakah jawaban “setuju” atau “tidak setuju” disebut yang disenangi, tergantung dari isi pertanyaan dan isi dari pernyataan yang disusun. Sewaktu menanggapi pertanyaan dalam skala Likert, responden menentukan tingkat persetujuan mereka terhadap suatu pernyataan dengan memilih salah satu dari pilihan yang tersedia. Contoh tingkat persetujuan sebuah pernyataan dapat dilihat di bawah ini.

- Sangat (tidak setuju / buruk / kurang sekali), memiliki skor 1
 - Tidak atau kurang (setuju / baik), memiliki skor 2
 - Netral / cukup, memiliki skor 3
 - Setuju / baik / suka, memiliki skor 4
 - Sangat (setuju / baik / suka), memiliki skor 5
4. Menghitung skor dari masing-masing individu pernyataan dengan mengali jumlah responden pada sebuah tingkat persetujuan dengan skor tingkat persetujuan. Setelah semua tingkat persetujuan sudah dikali, nilai semuanya dijumlahkan untuk mendapatkan total skor sebuah pernyataan.
5. Total skor tersebut dianalisis untuk mengetahui bagian mana yang sangat nyata batasannya antara skor tinggi dan skor rendah dalam skala total. Skor tertinggi didapatkan dari skor tertinggi tingkat persetujuan dikali dengan total responden. Sedangkan skor terendah didapatkan dari skor terendah tingkat persetujuan dikali dengan total responden.

6. Menentukan kriteria interpretasi yang berbentuk persentase. Antar kriteria memiliki interval yang didapatkan dengan membagi angka 100 dengan jumlah tingkat persetujuan yang ada. Contoh presentasi nilai jika intervalnya 20% yaitu sebagai berikut.

- Angka 0% – 19,99% = Sangat (tidak setuju / buruk / kurang sekali)
- Angka 20% – 39,99% = Tidak setuju / kurang baik
- Angka 40% – 59,99% = Cukup / netral
- Angka 60% – 79,99% = Setuju / baik / suka
- Angka 80% – 100% = Sangat (setuju / baik / suka)

7. Menghitung nilai interpretasi yang dapat ditentukan dengan menggunakan rumus sebagai berikut .

$$\text{Nilai interpretasi} = \frac{\text{Total Skor}}{\text{Total Skor Max}} \times 100 \quad \dots (2.3)$$

Rumus tersebut menghasilkan nilai berbentuk persen. Lalu nilai ini dicocokkan dengan kriteria interpretasi yang ada.

