BAB II

TINJAUAN PUSTAKA

2.1. Simulasi

Menurut Senersten (2010) Simulasi adalah sebuah bentuk model dari suatu masalah atau situasi yang bersifat tidak nyata yang bisa digunakan untuk mengajarkan seseorang untuk melakukan sesuatu (hlm 16). Dalam hal ini simulasi komputer menjadi contoh dari studi kasus tersebut. Menurut Mchaney (2009) Simulasi komputer biasa digunakan untuk menghindari atau mengurangi resiko kecelakaan proyek yang nantinya berlangsung dengan merubah atau membuat tatanan baru sistem tersebut (hlm 8). Contoh sederhana pengaplikasian simulasi komputer yang sudah ada saat ini adalah simulasi bangun kota. Simulasi juga mengenal istilah *Simulator*, yaitu sebuah alat simulasi yang dirancang untuk merepresentasikan suatu kondisi atau kejadian yang ada dari versi dunia nyata. Contoh dari alat simulasi ini seperti alat simulasi pesawat terbang untuk uji coba pilot pada saat melakukan tes penerbangan.

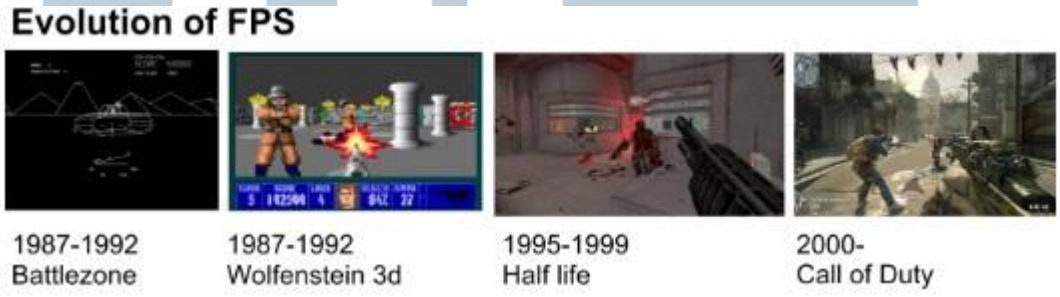
Senersten (2010) menjelaskan simulasi latihan tembak memiliki 5 batasan dalam perancangannya (hlm 27):

1. Perancangan model dari versi dunia nyatanya.
2. Mensimulasikan waktu dalam versi dunia nyatanya ke dalam simulasi latihan militer.

3. Melakukan riset konten untuk bisa mendapatkan karakter dalam perancangan simulasi militer atau latihan tembak tersebut.

4. Menyederhanakan isi dari perancangan simulasi yang rumit.
5. Mengevaluasi tingkat efektifitas dari simulasi tersebut ketika diuji kepada target audiensnya.

2.2. *Game* dan Simulasi *First Person Shooter*

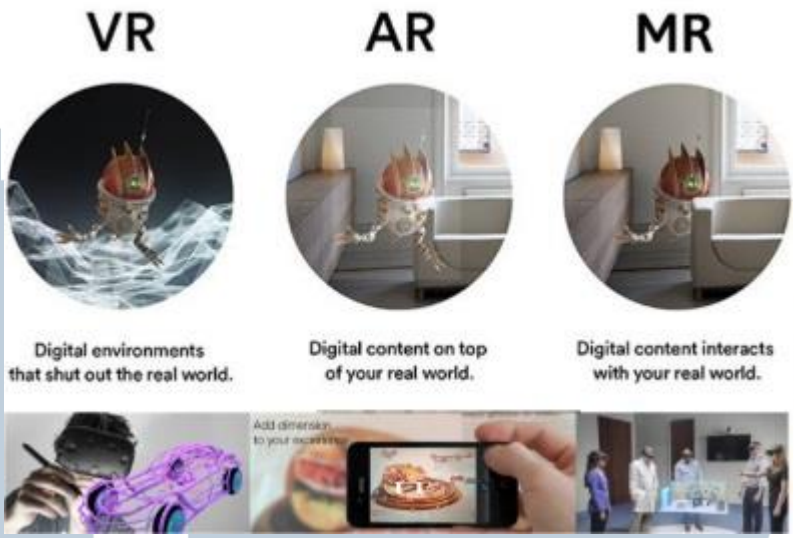


Gambar 2.1. Evolusi *Game First-Person Shooter* 20 Tahun Terakhir
(https://cdn.mos.cms.futurecdn.net/xFGzDECmgNoe59aLe2pHZX-970-80.jpg)

Menurut Elias (2009) *First-Person Shooter* merupakan *sub-genre* (sekunder) dalam genre *game* yang mengadaptasi tema tembak-menembak (*shooter*) dengan menggunakan sudut pandang mata karakter dalam *game* digital (hlm 9). *Sub-genre* pada *game first-person shooter* selalu melibatkan karakter utama dalam *game* digital untuk terlibat secara langsung dengan sebuah kejadian yang berlangsung dengan lingkungan sekitar lewat mata dari karakter pemain. Pada jamannya era *sub-genre* ini muncul, banyak konsep dan teori yang mendapatkan bahwa *genre first-person shooter* menjadi contoh dari bentuk *virtual reality* (realitas tidak nyata) dikarenakan faktor perspektif diambil dari mata karakter *game*. Dikarenakan faktor perspektif, *sub-genre* ini memerankan peran besar pada realisme dalam berbagai macam bentuk. Contoh sederhananya ketika dalam *game*

militer, pemain harus menggunakan helm perang dimana mata karakter pemain bisa melihat detail dari aset 3D helm seakan-akan pemain nyata yang menggunakan helm tersebut.

Virtual reality (realitas tidak nyata) merupakan *sub-genre* yang membuat peran *user* atau pemain dapat merasa seakan-akan terlibat secara langsung dengan apa yang terjadi di lingkungan sekitar simulasi digital . Karakter dalam simulasi digital bisa bergerak dan bertindak berdasarkan apa yang *player* lakukan, membuat *player* bisa menjelajahi berbagai struktur lingkungan *game* atau simulasi. Keputusan berada di tangan *user* untuk menjelajahi lingkungan *game* yang dirancang seperti labirin dengan memecahkan teka teki, dan membuka ruang *level* selanjutnya. Dikarenakan kedekatannya dengan realisme (kenyataan), batas dari simulasi dan kenyatan (*realism*) itu sendiri hanya terletak pada layar *game* atau simulasi digital tersebut. Namun dalam segi *virtuality* (bentuk tidak nyata), semakin tidak *virtual* (tidak nyata) sebuah *game* atau simulasi *first-person shooter*, semakin kuat kedekatannya dengan realisme. Contoh sederhana seperti kurang atau tidak ada sama sekali tampilan tatap muka (*user interface*) pada *game* dan simulasi digital.



Gambar 2.2. Contoh *Virtual Reality*, *Augmented Reality*, dan *Mixed Reality* (https://res.cloudinary.com/madimages/image/fetch/e_sharpen:100,q_auto:eco,fl_progressive:semi/https://www.mobileappdaily.com/public/uploads/mad_488689329f.png)

2.3. *User Experience* Dalam *Game* dan Simulasi *First-Person Shooter*

Bernhaupt (2010) menjelaskan istilah *User Experience* masih berkaitan kuat dengan peran *Human Computer Interface* dalam memahami alur kerja pembuatan dan penggunaan produk secara digital (hlm 3). *User Experience* dalam *game* dievaluasi dengan berbagai macam konsep, seperti:

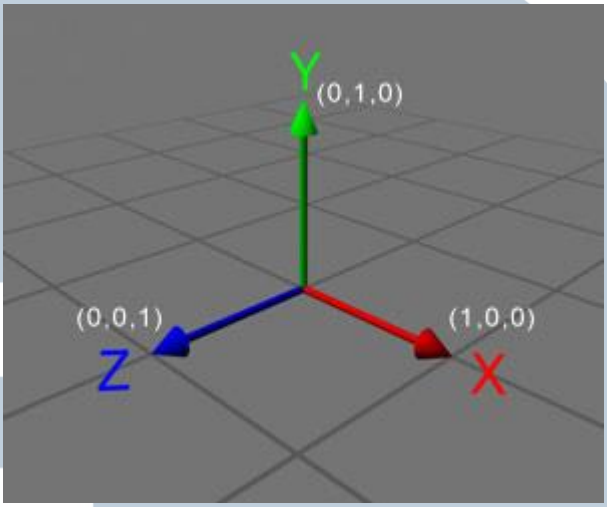
- *playability* (bisa dimainkan),
- *engagement* (keterlibatan pemain),
- *fun* (faktor menyenangkan),
- *flow* (alur *game*)

Secara keseluruhan *User Experience* memfokuskan diri pada persepsi orang, apa yang dirasakan dan apa yang diharapkan dari sebuah produk digital.

Perancangan *User Experience* dalam *game* digital melibatkan 6 tahapan prosedur perancangan.

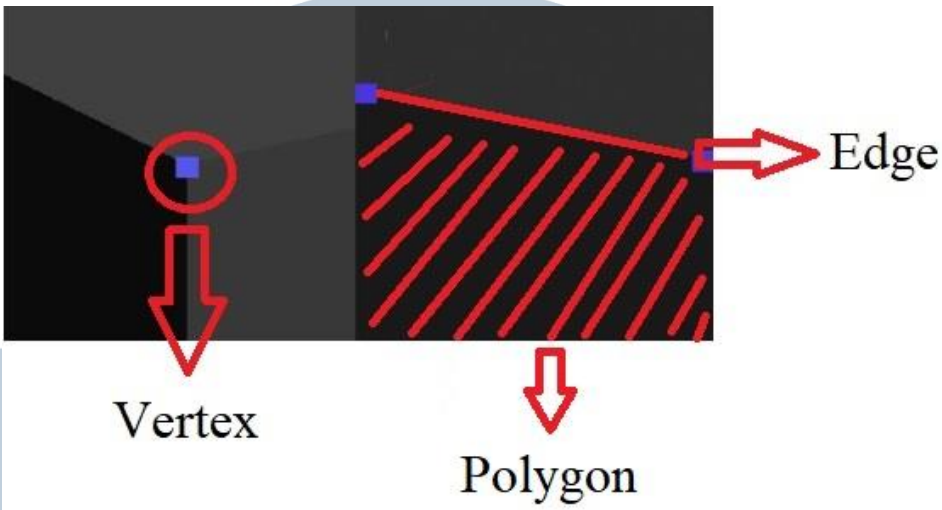
1. Pertama konsep, mengemukakan sebuah gagasan dan ide untuk memberikan gambaran sebuah *game*.
2. Kedua pra-produksi, berfokus pada arahan *timeline* (alur waktu) pengerjaan proyek *game* mulai dari pengarahan *style* aset, perencanaan produksi, perancangan teknis secara laporan.
3. Ketiga pembuatan purwarupa (*prototype*), untuk membuat gambaran produk digital secara kasar (belum terselesaikan) untuk memberikan gambaran visual di fase final.
4. Keempat produksi, proses yang dilakukan selama bertahun-tahun dalam mengeksekusi proyek dari yang sudah direncanakan pada saat fase pra-produksi.
5. Kelima fase *Alpha* dan *Beta*, fase untuk melakukan tes produk (*game*) untuk mengetahui terdapat masalah teknis dalam proses perancangan (*bug*).
6. Keenam fase *gold* dan pasca produksi, fase terakhir dimana produk *game* digital sudah dirilis dan mengalami proses evaluasi dari pemain dan pengguna untuk mengetahui baik dan buruknya produk.

2.4. Pembahasan Model 3D Dalam Simulasi



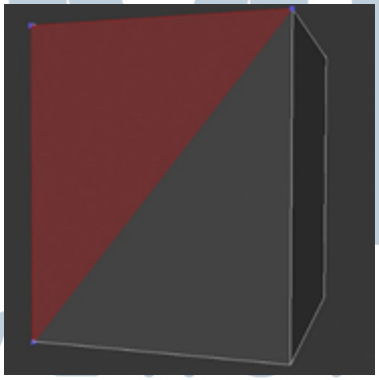
Gambar 2.3. Koordinat X, Y, dan Z Dalam *Software* 3D
(http://www.andynicholas.com/wp-content/uploads/2009/10/AxisIcon.jpg)

Sebuah simulasi digital 3D, terdapat istilah *vertex*, *edge*, dan *polygon*. V*ertex* adalah sebuah titik tunggal dari sebuah model 3D yang diindikasikan lewat koordinat X, Y, dan Z. Untuk *edge*, Gahan (2012) menyebutkan *edge* adalah sebuah garis yang menghubungkan dua *vertex*. *Polygon* adalah sebuah permukaan model 3D yang dibuat dari 4 *vertex* atau *edge*. Dari 3 hal ini sudah menjelaskan bentuk dasar dari sebuah model 3D (hlm 6-8).

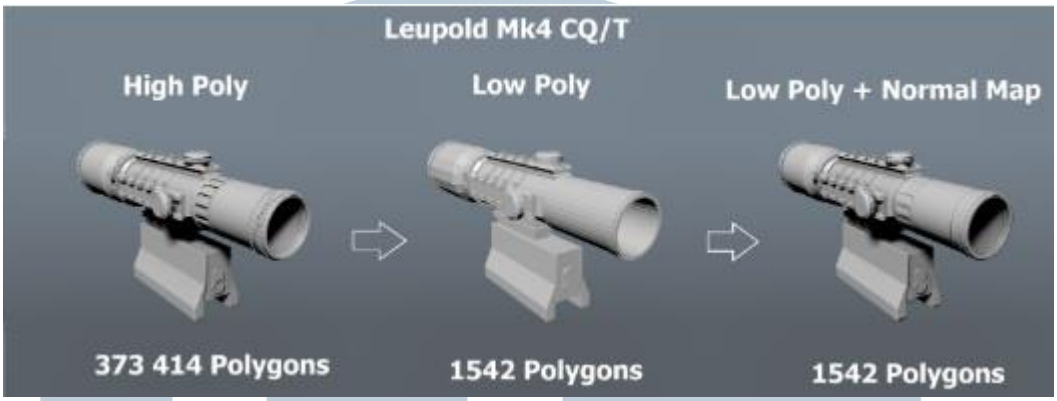


Gambar 2.4. Contoh dari *Vertex* dan *Edge*
(*Max Modelling for Games*, 2009)

Meskipun *polygon* adalah apa yang dilihat dari sebuah model 3D, namun model 3D dalam sebuah aset simulasi atau *game* lebih berfokus pada *faces* atau bisa disebut *tris*. *Faces* terbentuk ketika tiga *vertex* atau titik tunggal membentuk sebuah *edge* atau garis sambung antara ketiga *vertex* tersebut.



Gambar 2.5. Contoh *Faces* dalam Model 3D
(*Max Modelling for Games*, 2009)



Gambar 2.6. Contoh Model 3D *High Polygon* dan *Low Polygon* untuk *Game*
(http://1.bp.blogspot.com/-peONqYXn_aE/Vk7cly2UiNI/AAAAAAAABL8/1X-_-H0BjjU/s1600/266963_367010970057889_1979841288_o.png)

Dalam sebuah perancangan model 3D *game*, seorang 3D *modeler* harus memperhatikan model 3D *low-poly* pada asetnya. Gahan (2009) dalam industri *games* penggunaan tekstur dalam model 3D lebih diutamakan atas faktor optimalisasi (hlm 1). Hal ini menjelaskan bahwa model 3D dalam *game* harus bersifat *low-poly*. Namun dari model 3D *low-poly* tersebut akan ditambahkan tekstur dan *mapping* yang menambah kesan model model 3D *low polygon* seperti *high polygon*. Hal ini dilakukan agar model 3D *game* tersebut berjalan dengan optimal dalam sebuah *hardware*.

Gambar 2.7. Contoh Model 3D Low-Poly Box Dengan Tekstur
(*Max Modelling for Games*, 2009)

2.4.1. Level of Detail

Sebuah *game* dan simulasi mengenal istilah *level of detail*, disingkat dengan *LOD*. Menurut Ahearn (2012) *Level of detail* adalah aset 3 Dimensi (3D) yang bersifat *low polygon* dan memiliki kesan seperti model 3D *high polygon* (hlm 9). Hal ini perlu dilakukan untuk bisa mengoptimalisasi sebuah model 3D yang ada untuk dimasukkan ke sebuah *game engine*. Efisiensi visual dalam sebuah *game* sangat penting agar mencapai *visual style* dalam *game* dan simulasi yang diinginkan dengan pencapaian sisi teknis yang optimal.

Gambar 2.8. Contoh LOD dalam Sebuah Model 3D
(*3D Game Environment*, 2012)

Dalam *Level of Detail*, tekstur juga merupakan salah satu faktor dari kunci optimisasi. Ahearn (2008) menjelaskan tekstur akan digunakan untuk memberikan material dan warna pada objek 3D (hlm 8). Tekstur yang bagus adalah yang bisa membuat model 3D *low polygon* menjadi seakan-akan *high polygon*. Untuk bisa mengoptimalisasi sebuah kerja *game engine* dikenal dengan *texture packing* yaitu cara agar model 3D yang memiliki tekstur yang sama dimasukkan dalam satu

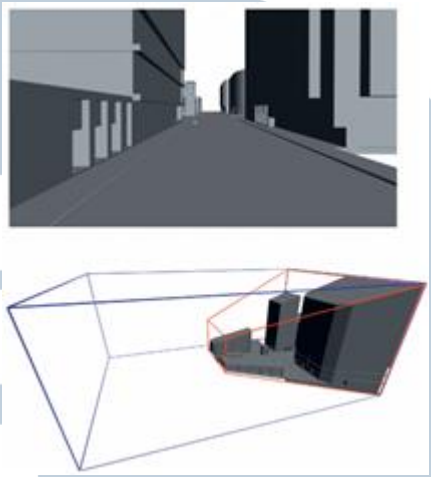
template tekstur. Contohnya adalah salah satu lokasi *game* di hutan yang memiliki ciri-ciri model 3D tumbuhan yang sama dengan yang lainnya.

Gambar 2.9. Contoh Tekstur
(*3D Game Environment*, 2012)

Level of Detail mengenal istilah *Collision Object* dalam pembuatan aset visual permainan. Ahearn (2008) menambahkan *Collisions* adalah sebuah bentuk objek tak tampak yang bertujuan sebagai alat bertumbuknya karakter dalam *game* (hlm 26). Jika suatu aset 3D tidak memiliki *Collisions*, maka aset tersebut dapat tertembus oleh karakter *game* saat bertumbukan. Jadi *Collision Object* dalam *game* tidak akan terlihat pada saat jalannya permainan. Pembuatan *Collisions* berasal dari objek 3D yang transparan, perlu dibuat model 3D *low-poly* yang bertujuan untuk optimisasi tersebut. Bentuk dari *Collisions* itu sendiri bisa berupa kubus atau silinder yang hampir menyerupai bentuk aset 3D tersebut.

Gambar 2.10. Contoh dari Collisions yang Bertumbukan dengan Objek Karakter (*3D Game Environment*, 2012)

Dalam perancangan aset 3D *game environment*, sangat perlu diperhatikan optimisasi *polygon* dari suatu *environment*. Ahearn (2009) menyebutkan ketika seorang *3D Game Artist* membuat model 3D dia bisa menghapus atau menghilangkan bagian-bagian *polygon* yang tidak perlu karena tidak akan terlihat dalam *game* (hlm 29). Tujuannya agar tidak *lag*.



Gambar 2.11. Contoh dari Optimisasi Polygon
(*3D Game Environment*, 2012)

Pada contoh gambar 2.11 menunjukkan bahwa perancangan aset 3D *environment* dalam sebuah *game* hanya dibatasi dari apa yang dilihat oleh objek utama dalam *game*. Objek utama yang dimaksud bisa berupa karakter utama atau kamera *game* yang sedang memperlihatkan area 3D *environment* tersebut. Jika sebuah *environment* tersebut tidak perlu dibuat atau hanya terlihat dari jarak jauh, maka model 3D bisa dibuat *low-poly* untuk menunjang optimisasi tersebut.

Sebuah lingkungan (*environment*) dalam sebuah *game* juga harus memperhatikan ukuran luas dari wilayah permainannya. Teknologi yang akan menjalankan *game* tersebut harus memperhatikan apa yang bisa dilakukan dan apa yang tidak bisa dilakukan dalam area *game* tersebut. Contoh dari kasus ini, karakter dalam pemain akan dimasukkan yang berada di *game world* tersebut bisa memiliki keterbatasan ketika sedang menjelajah. Salah satu contohnya pemain tidak bisa mengelilingi seluruh pelosok *game world* yang ada dikarenakan ada tembok tidak tembus pandang berupa *collisions object* yang ditaruh di salah satu

area tersebut. Hal ini biasa dilakukan oleh *game developer* untuk membatasi kemampuan visual dalam satu *platform hardware* yang digunakan untuk *game*-nya. Contoh sederhananya adalah adanya tekstur model 3D yang tidak beresolusi tinggi atau sangat *low-poly* yang akan menurunkan kualitas *game* tersebut. Setiap *game developer* harus memperhatikan dari sisi mana saja area-area *game* atau simulasi tersebut yang bisa berpotensi untuk dijelajahi atau tidak.

Dalam proses pembuatan aset 3D lingkungan (*environment*) *game*, terdapat istilah *Occlusion* dan *Culling*. Ahearn (2009) menyebutkan *Occlusion* artinya suatu objek yang menutupi objek lainnya dalam satu jarak pandang sedangkan *Cull* sendiri lebih kepada menyingkirkan pada suatu hal atau grup (objek) yang tidak terlihat (hlm 28). Dalam hal pembuatan aset 3D lingkungan (*environment*) *game*, kedua arti kata ini menjelaskan tentang bagaimana proses sebuah *game* hanya memperlihatkan bagian-bagian terpenting dalam objek 3D antara yang terlihat dan tidak terlihat. Dalam prosesnya lebih merujuk pada efisiensi ketika *game* dan simulasi digital tersebut sedang *rendering* sebuah aset 3D lingkungan (*environment*) dalam permainannya. Syaratnya bila suatu objek tidak diperlihatkan oleh kamera *game*-nya atau tertutupi oleh objek lain di depan kamera *game*, maka dihilangkan untuk efisiensi dalam proses *rendering* di *game engine*.

Gambar 2.12. Contoh *Occlusion Culling* dalam *Game Engine Unity*

(https://www.youtube.com/watch?v=UiKlKLt9zkM)

2.4.2. Integrasi Aset ke dalam *Game Engine*

Gambar 2.13. Contoh logo *game engine Unity* dan *Unreal Engine*

(https://upload.wikimedia.org/wikipedia/commons/1/19/Unity_Technologies_logo.svg &
http://indiemegabooth.com/wp-cargo/uploads/2016/09/UNREAL-ENGINE-4-LOGO6.jpg)

Setiap *game* mempunyai sebuah *game engine*. Menurut Gregory (2009) *game engine* adalah sebuah *software* yang digunakan oleh *game developer* untuk menggabungkan aset 3D dan *sound effect* dalam *game* (hlm 11). Penggunaan mesin *game* digital telah banyak dikenal pada era 1990an. Fungsi dari penggunaan

mesin *game* ini adalah untuk bisa membantu menggabungkan seluruh aset yang ada dalam permainan. Setiap *game engine* memiliki masing-masing keunggulan untuk bisa memperlihatkan kualitas visual yang bagus dan efisien.

Setiap *game engine* selalu memiliki model 3D properti dalam permainannya. Properti dalam *game engine* berfungsi untuk bisa mengisi ruang yang kosong dalam sebuah *game* ketika sedang dibuat dalam *game engine* tersebut. Menurut Watkins (2016) Tantangan dalam penataan model 3D properti adalah peletakan properti sama seperti dengan dunia nyatanya (hlm 286). Penempatan objek-objek properti dalam sebuah *game* selalu berasal dari konsep gambar yang telah dibuat yang akan menjadi titik acuan untuk perancangan aset 3D *environment game* tersebut. Contoh sederhana dari keselarasan antara penempatan model 3D properti dalam *game engine* dengan tema *environment*-nya seperti tempat bar minum dengan model 3D properti seperti botol minum, meja bar, dan lainnya.

Gambar 2.14. Contoh Model 3D Properti yang Selaras dengan Tema Environment Permainan Di Tempat Bar Minum
(*Procedural Content Generation for Unity Game Development*, 2016)

Integrasi model 3D dalam perancangan aset 3D *environment* ini akan dilakukan dalam *game engine* bernama *Unity* khususnya *Unity 3D*. Menurut Creighton (2010) menyebutkan bahwa *Unity 3D* adalah *game engine* yang ditujukan kepada *game developer* pemula dalam melakukan proses integrasi aset 3D dalam *game* (hlm 7).

Eng (2015) menjelaskan integrasi aset ke dalam *Unity* 3D adalah dengan membuat *folder* terlebih dahulu untuk proyek *game* 3D, setelah itu memasukkan berkas (*file*) 3D ke dalam berkas folder (*folder*) di proyek tersebut lalu digeser aset 3D tersebut ke dalam *user interface* berbentuk visualisasi dalam *game engine* (hlm 106). Pada *Unity 3D*, integrasi asetnya dapat dikatakan mudah hanya dengan menggeser file aset ke *user interface game engine Unity*. Kemudahan integrasi aset dalam *game engine Unity 3D* ini membuat banyak sekali para pembuat *game* yang masih pemula untuk menggunakan *game engine* ini. Hasil akhir visual dalam *game* biasanya mencakup dari gaya visual yang realis hingga kartunis.

Gambar 2.15. Contoh Bagian *User Interface* Utama dari *Game Engine Unity* 3D (Building a Game with Unity and Blender, 2011)

Gambar 2.16. Gaya Visual *Stylish* dalam *Unity*
(https://www.youtube.com/watch?v=1arUX-aMLqQ)

Gambar 2.17. Gaya Visual *Realistic* dalam *Unity*
(https://www.youtube.com/watch?v=6oo293kIGPQ)