BAB II
LANDASAN TEORI

II.1 *Augmented Reality*

Augmented Reality (AR) adalah penggabungan benda-benda nyata dan maya dilingkungan nyata, berjalan secara interaktif dalam waktu nyata, dan terdapat integrasi antar benda dalam tiga dimensi, yaitu benda maya terintegrasi dalam dunia nyata [1]. Beberapa metode yang dikembangkan dalam teknologi AR, diantaranya *Gesture Based*, *Occlusion Based*, *Marker Based* ataupun *Markerless*.

1. Metode *Gesture Based*

   Suatu sistem komputasi yang terjadi karena adanya interaksi antara manusia dengan sumber digital tanpa menggunakan perangkat *input* seperti *keyboard*, *mouse*, kontroler *game*.

   a. Kelebihan : banyaknya variasi masukan yang dapat dilakukan, lebih mudah dilakukan, lebih alami, lebih interaktif dan membutuhkan proses pembelajaran yang relatif lebih singkat.
   b. Kekurangan : untuk metode ini kurang cocok bila digunakan untuk penampilan objek *virtual* karena metode ini difokuskan hanya untuk mendeteksi suatu gerakan.

2. Metode *Occlusion Based*

   Merupakan metode yang hanya mendukung interaksi dua dimensi dan membutuhkan jumlah *marker* yang cukup banyak.

   a. Kelebihan : menghasilkan sistem AR yang lebih *real time* karena jeda waktu menampilkan *video* lebih cepat dan metode ini cenderung memiliki nilai komputasi yang lebih rendah dibandingkan dengan metode *gesture based*.
   b. Kekurangan : pada metode ini memiliki kekurangan seperti, interaksi yang didukung hanya dua dimensi dan banyaknya *marker* yang dipakai.

3. Metode *Marker Based*

   Merupakan suatu ilustrasi hitam dan putih persegi dengan batas hitam tebal dan latar belakang yang berwarna putih.

a. Kelebihan : metode ini sudah dikembangkan oleh banyak *library* sehingga begitu mudah untuk penggunaan nya, metode ini juga sudah banyak digunakan dalam kehidupan sehari – hari.
b. Kekurangan : metode ini kurang begitu menarik karena *marker* yang digunakan hanya berupa persegi panjang yang memiliki warna hitam dan putih.

4. Metode *Markerless*

Suatu *marker* yang digunakan untuk mendeteksi objek, tapi *marker* yang digunakan berbeda dengan *marker based tracking* yang masih menggunakan *marker* persegi berwarna hitam dan putih. *Markerless* menggunakan fitur alami (*natural features*) dari gambar atau objek yang akan dilacak.

a. Kelebihan : Dalam metode ini membuat *marker* menjadi lebih alami, metode ini juga memiliki beberapa teknik yang dipakai untuk pembuatan nya seperti *face tracking, 3d object tracking* dan *motion tracking*.
b. Kekurangan : Dalam implementasinya metode ini menggunakan banyak *feature point* sebagai penandanya, sehingga bila akan menampilkan objek *virtual*, *feature point* tersebut harus terdeteksi sepenuhnya oleh kamera. Jika tidak maka objek *virtual* yang dihasilkan akan tidak cukup baik.

Kesimpulan dari perbandingan metode untuk *augmented reality* diatas adalah metode *markerless* lebih cocok untuk digunakan dalam penelitian ini karena metode *markerless* membuat *marker* yang dipakai menjadi lebih alami dan untuk proses pembuatannya metode ini memiliki beberapa teknik yang dapat digunakan.

Augmented reality dapat diklasifikasikan menjadi dua berdasarkan ada tidaknya pengguaan *marker* yaitu :

1. *Marker Augmented Reality*

Sebuah metode yang memanfaatkan *marker* yang biasanya berupa ilustrasi hitam dan putih berbentuk persegi atau lainnya dengan batas hitam tebal dan latar belakang putih. Melalui posisi yang dihadapkan pada sebuah kamera komputer atau *smartphone*, maka komputer atau *smartphone* akan melakukan proses menciptakan sebuah dunia *virtual* 3D yaitu titik (0,0,0) dan sumbu yang terdiri dari

x, y dan z. *Marker based tracking* ini sudah lama dikembangkan sejak tahun 1980 an. *Library* atau SDK yang dapat digunakan untuk teknik ini adalah ARToolkit.

2. *Markerless Augmented Reality*

Salah satu metode *Augmented Reality* yang saat ini sedang berkembang adalah metode *Markerless Augmented Reality*. Dengan metode ini pengguna tidak perlu lagi menggunakan sebuah *marker* untuk menampilkan objek 3D atau yang lainnya. Sekalipun dinamakan dengan *markerless* namun aplikasi tetap berjalan dengan melakukan pemindaian terhadap objek, namun ruang lingkup yang dipindai lebih luas dibanding dengan *Marker Based Tracking*. Seperti yang saat ini dikembangkan oleh perusahaan *Augmented Reality* terbesar di dunia Total Immersion. Adapun beberapa teknik yang digunakan dalam *Markerless Augmented Reality* adalah sebagai berikut :

a. *Face Tracking*

Dengan menggunakan algoritma yang banyak dikembangkan, komputer dapat mengenali wajah manusia secara umum dengan cara mengenali posisi mata, hidung, dan mulut manusia, kemudian akan mengabaikan objek-objek lain di sekitarnya seperti pohon, rumah, dan benda-benda lainnya.

b. *3D Object Tracking*

Berbeda dengan *Face Tracking* yang hanya mengenali wajah manusia secara umum, teknik 3D *Object Tracking* dapat mengenali semua bentuk benda yang ada di sekitar, seperti mobil, meja, televisi, dan lain-lain.

c. *Motion Tracking*

Pada teknik ini komputer dapat menangkap gerakan, *Motion Tracking* telah mulai digunakan secara *ekstensif* untuk memproduksi film-film yang mencoba menyimulasikan gerakan. Contohnya pada film Avatar, dimana James Cameron menggunakan teknik ini untuk membuat film tersebut dan menggunakannya secara *real-time*.

II.2 *Marker*

Marker merupakan bentuk *marker* yang sering digunakan dalam teknologi *augmented reality*. *Marker* atau *fiducial marker* bebentuk kotak dengan pola warna hitam dan putih, didalamnya terdapat gambar yang telah dikenali sebelumnya.

Marker ini memiliki fungsi untuk menampilkan objek *virtual* nya. Berikut contoh dari *marker* atau *fiducial marker* pada gambar 2.1



Gambar II. 1 *Fiducial marker*

II.3 *Markerless*

Menurut Madden *markerless* adalah AR yang digunakan untuk melacak objek yang ada di dunia nyata tanpa *marker* yang spesial. *Markerless* sangat berbeda dengan *fiducial marker* dan *markerless* ini melakukan proses pelacakan bergantung pada *natural feature-tracking*. *Markerless* dalam *library* ARToolKit dapat melakukan proses *tracking* pada sebuah foto atau dokumen dengan bidang berbentuk persegi panjang atau bidang datar, proses *tracking* terhadap bidang datar tersebut diharuskan dapat terdeteksi seluruhnya oleh kamera *webcam* agar hasil yang didapat terlihat cukup baik, karena bila *marker* hanya terdeteksi sebagian saja atau hanya dapat terlihat 50% maka objek yang dihasilkan tidak stabil. Tidak stabil di sini objek yang dihasilkan bisa muncul dengan tampilan yang tidak cukup baik ataupun tidak muncul sama sekali. AR *markerless* melakukan pelacakan dengan objek yang sudah ada sebelumnya yang kemudian dijadikan *image target* atau gambar yang akan dideteksi untuk menampilkan objek *virtual*. Seperti yang saat ini dikembangkan oleh perusahaan *Augmented Reality* terbesar di dunia Total Immersion dan Qualcomm, mereka telah membuat berbagai macam teknik *Markerless Tracking* sebagai teknologi andalan mereka, seperti *Face Tracking*, 3D *Object Tracking*, dan *Motion Tracking*.

a. *Face Tracking*

Dengan menggunakan algoritma yang mereka kembangkan, komputer dapat mengenali wajah manusia secara umum dengan cara mengenali posisi mata, hidung, dan mulut manusia, kemudian akan mengabaikan objek-objek lain di sekitarnya seperti pohon, rumah, dan benda-benda lainnya. Teknik ini pernah digunakan di Indonesia pada Pekan Raya Jakarta 2010 dan *Toy Story 3 Event*.

b. *3D Object Tracking*

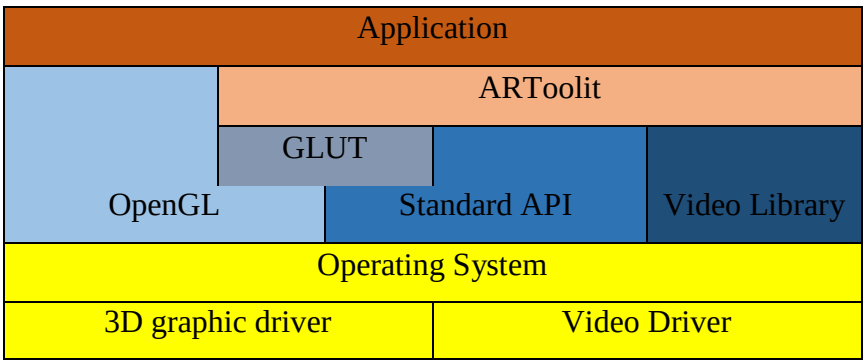
Berbeda dengan *Face Tracking* yang hanya mengenali wajah manusia secara umum, teknik *3D Object Tracking* dapat mengenali semua bentuk benda yang ada disekitar, seperti mobil, meja, televisi, dan lain-lain.

c. *Motion Tracking*

Pada teknik ini komputer dapat menangkap gerakan, *Motion Tracking* telah mulai digunakan secara *ekstensif* untuk memproduksi film-film yang mencoba mensimulasikan gerakan. Contohnya pada filmAvatar, di mana James Cameron menggunakan teknik ini untuk membuat film tersebut dan menggunakannya secara *realtime.*

II.4 ARToolkit

ARToolkit merupakan *software tookit* layaknya GLUT. Di dalamnya terdiri dari *predefined functions* yang bisa kita panggil (*call*) dalam mengembangkan aplikasi AR. ARToolKit menggunakan OpenGL untuk *rendering*, GLUT untuk aspek *windows*/*event handler*. API ARToolKit ditulis dalam bahasa C dan disertakan pula beberapa contoh yang bisa dijadikan acuan dalam mengembangkan aplikasi AR. Gambar di bawah mengilustrasikan hubungan antara aplikasi yang dibuat dengan ARToolKit dan *library* yang diperlukan.



Gambar II. 2 ARTookit *Architecture*

ARToolKit pertama kali dirilis secara terbuka di SIGGRAPH pada tahun 1999. Sedangkan untuk *open source* sendiri dirilis pada tahun 2001 dan dikomersialkan oleh ARToolworks, ARToolKit telah diunduh hampir 1 juta kali. Namun setelah diakuisisi oleh daqri, *library* ARToolkit yang sebelumnya tersedia dengan lisensi berbayar, maka kini telah dirilis secara gratis dibawah lisensi LGPL

II.5 Tipe Data

Artoolkit memanipulasi beberapa *variable* yang berbeda. ARToolKit menggunakan *format image* yang berbeda untuk modul yang berbeda. Gambar dibawah mengilustrasikan beberapa *format* yang didukungnya. Beberapa *format* hanya tersedia pada beberapa *platform* dan *hardware* yang spesifik.



Video Format (*Camera Format*)	*Tracking Format*	*Display Format*	*Rendered Format*
RGB24/RGB32	RGBA/AGBR	RGBA/BGRA	RGB24
YUV/VUY	RGB/BGR	RGB/BGR	RGB555
YUV422/YUV422P			YUV
YUV420/YUV420P			
YUV411			

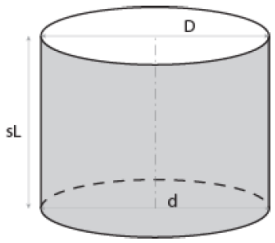
Gambar II. 3 *Data Flow* ARToolkit

II.6 *Tracking Cylindrical Object*

Objek silinder digunakan untuk mendeteksi dan melacak gambar yang digulung menjadi silinder. Objek silinder juga mendukung untuk deteksi dan pelacakan secara sebagian *marker* saja.

Objek silinder ini sebenarnya sudah tersedia di *library* Vuforia SDK dengan *target manager* yang telah diketahui sebelumnya. Sehingga dalam penelitian ini *tracking cylindrical object* akan dicoba digunakan pada *library ARToolkit* dengan tujuan sebagai pengembangan *library*. Untuk pengembangan nya akan menggunakan metode *markerless* sebagai pembacaan terhadap *marker* nya, Untuk itu bagaimana caranya agar *marker* yang dilacak tetap dapat menampilkan objek

yang cukup baik walaupun deteksi *marker* terhadap kamera hanya terlihat sebagian saja dikarenakan memiliki bidang yang tidak datar. Berikut merupakan contoh dari satu kasus object silinder pada gambar II.4.



Gambar II. 4 *Cylindrical Object*

Keterangan :

D = *Top Diameter* (Diameter Atas)

d = *Bottom Diameter* (Diameter Bawah)

sL = *Side Length* (Panjang Sisi)

II.7 C *Sharp (C#)*

C# merupakan sebuah bahasa pemrograman yang berorientasi objek yang dikembangkan oleh Microsoft sebagai bagian dari inisiatif kerangka .NET Framework. Bahasa pemrograman ini dibuat berbasiskan bahasa C++ yang telah dipengaruhi oleh aspek-aspek ataupun fitur bahasa yang terdapat pada bahasa-bahasa pemrograman lainnya seperti Java, Delphi, Visual Basic, dan lain-lain) dengan beberapa penyederhanaan. Menurut standar ECMA-334 C# *Language Specification*, nama C# terdiri atas sebuah huruf Latin C yang diikuti oleh tanda pagar yang menandakan angka #. Tanda pagar # yang digunakan memang bukan tanda kres dalam seni musik, dan tanda pagar # tersebut digunakan karena karakter kres dalam seni musik tidak terdapat di dalam *keyboard* standar.

Standar *European Computer Manufacturer Association* (ECMA) mendaftarkan beberapa tujuan *desain* dari bahasa pemrograman C#, sebagai berikut:

1. Bahasa pemrograman C# dibuat sebagai bahasa pemrograman yang bersifat bahasa pemrograman *general-purpose* (untuk tujuan jamak), berorientasi objek, moderen, dan sederhana.

2. Bahasa pemrograman C# ditujukan untuk digunakan dalam mengembangkan komponen perangkat lunak yang mampu mengambil keuntungan dari lingkungan terdistribusi.
3. Portabilitas *programmer* sangatlah penting, khususnya bagi *programmer* yang telah lama menggunakan bahasa pemrograman C dan C++.
4. Dukungan untuk internasionalisasi (*multi-language*) juga sangat penting.
5. C# ditujukan agar cocok digunakan untuk menulis program aplikasi baik dalam sistem *klien-server* (*hosted system*) maupun sistem *embedded* (*embedded system*), mulai dari perangkat lunak yang sangat besar yang menggunakan sistem operasi yang canggih hingga kepada perangkat lunak yang sangat kecil yang memiliki fungsi-fungsi terdedikasi.
6. Meskipun aplikasi C# ditujukan agar bersifat 'ekonomis' dalam hal kebutuhan pemrosesan dan memori komputer, bahasa C# tidak ditujukan untuk bersaing secara langsung dengan kinerja dan ukuran perangkat lunak yang dibuat dengan menggunakan bahasa pemrograman C dan bahasa rakitan.
7. Bahasa C# harus mencakup pengecekan jenis (*type checking*) yang kuat, pengecekan larik (*array*), pendeteksian terhadap percobaan terhadap penggunaan *Variabel-variabel* yang belum diinisialisasikan, portabilitas kode sumber, dan pengumpulan sampah (*garbage collection*) secara otomatis.

Pada akhir dekade 1990-an, Microsoft membuat program Microsoft Visual J++ sebagai sebuah langkah percobaan untuk menggunakan Java di dalam sistem operasi Windows untuk meningkatkan antarmuka dari Microsoft *Component Object Model* (COM). Akan tetapi, akibat masalah dengan pemegang hak cipta bahasa pemrograman Java, *Sun Microsystems*, Microsoft pun menghentikan pengembangan J++, dan beralih untuk membuat pengganti J++, kompilernya dan mesin *virtual* nya sendiri dengan menggunakan sebuah bahasa pemrograman yang bersifat *general-purpose*. Untuk menangani proyek ini, Microsoft merekrut Anders Helsberg, yang merupakan mantan karyawan Borland yang membuat bahasa Turbo Pascal, dan Borland Delphi, yang juga mendesain *Windows Foundation Classes* (WFC) yang digunakan di dalam J++. Sebagai hasil dari usaha tersebut, C# pun pertama kali diperkenalkan pada bulan Juli 2000 sebagai sebuah bahasa

pemrograman modern berorientasi objek yang menjadi sebuah bahasa pemrograman utama di dalam pengembangan di dalam *platform* Microsoft .NET Framework.

Pengalaman Helsberg sebelumnya dalam pendesain bahasa pemrograman seperti Visual J++, Delphi, Turbo Pascal) dengan mudah dilihat dalam sintaksis bahasa C#, begitu pula halnya pada inti *Common Language Runtime* (CLR). Dari kutipan atas *interview* dan makalah-makalah teknisnya ia menyebutkan kelemahan-kelemahan yang terdapat pada bahasa pemrograman yang umum digunakan saat ini, misalnya C++, Java, Delphi, ataupun Smalltalk. Kelemahan-kelemahan yang dikemukakannya itu yang menjadi basis CLR sebagai bentukan baru yang menutupi kelemahan-kelemahan tersebut, dan pada akhirnya mempengaruhi *desain* pada bahasa C# itu sendiri. Ada kritik yang menyatakan C# sebagai bahasa yang berbagi akar dari bahasa-bahasa pemrograman lain. Fitur-fitur yang diambilnya dari bahasa C++ dan Java adalah *desain* berorientasi objek, seperti *garbage collection*, *reflection*, akar kelas (*root class*), dan juga penyederhanaan terhadap pewarisan jamak (*multiple inheritance*). Fitur-fitur tersebut di dalam C# kini telah diaplikasikan terhadap iterasi, properti, kejadian (*event*), metadata, dan konversi antara tipe-tipe sederhana dan juga objek.

C# didisain untuk memenuhi kebutuhan akan sintaksis C++ yang lebih ringkas dan *Rapid Application Development* yang 'tanpa batas' (dibandingkan dengan RAD yang 'terbatas' seperti yang terdapat pada Delphi dan Visual Basic) [5].

II.8 JavaScript

JavaScript adalah bahasa pemrograman tingkat tinggi dan dinamis. JavaScript populer di internet dan dapat bekerja di sebagian besar penjelajah web populer seperti Internet Explorer (IE), Mozilla Firefox, Netscape dan Opera. Kode JavaScript dapat disisipkan dalam halaman web menggunakan tag SCRIPT.

JavaScript pertama kali dikembangkan oleh Brendan Eich dari Netscape di bawah nama Mocha, yang nantinya namanya diganti menjadi LiveScript, dan akhirnya menjadi JavaScript. *Navigator* sebelumnya telah mendukung Java untuk lebih bisa dimanfaatkan para programmer yang non-Java. Maka dikembangkanlah

bahasa pemrograman bernama LiveScript untuk mengakomodasi hal tersebut. Bahasa pemrograman inilah yang akhirnya berkembang dan diberi nama JavaScript, walaupun tidak ada hubungan bahasa antara Java dengan JavaScript. JavaScript bisa digunakan untuk banyak tujuan, misalnya untuk membuat efek *rollover* baik di gambar maupun teks, dan yang penting juga adalah untuk membuat AJAX. JavaScript adalah bahasa yang digunakan untuk AJAX [6].

II.9 Unity 3D

Unity 3D adalah sebuah ekosistem dari pengembangan *game*, mesin *render* yang terintegrasi penuh dengan satu set peralatan yang intuitif dan alur kerja yang cepat untuk menciptakan konten 3D dan 2D yang interaktif, mudah dipublikasikan ke berbagai *platform*, aset tersedia di *asset store*, dan memiliki komunitas untuk berbagi pengetahuan.

Adapun fitur-fitur yang dimilik oleh Unity 3D antara lain sebagai berikut:

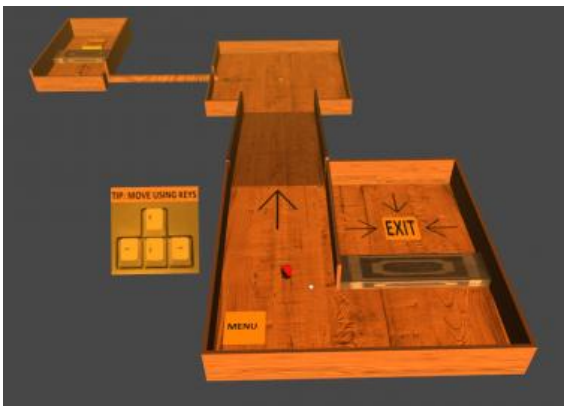
1. *Integrated development environment* (IDE) atau lingkungan pengembangan terpadu
2. Penyebaran hasil aplikasi pada banyak *platform*.
3. *Engine grafis* menggunakan Direct3D (Windows), OpenGL (Mac, Windows), OpenGL ES (iOS), and *proprietary* API (Wii).
4. *Game Scripting* melalui Mono. *Scripting* yang dibangun pada Mono, implementasi open source dari NET Framework. Selain itu Pemrogram dapat menggunakan UnityScript (bahasa kustom dengan sintaks JavaScriptinspired), bahasa C # atau Boo (yang memiliki sintaks *Python-inspired*).

Mesh merupakan bentuk dasar dari obyek 3D. Pembuatan *mesh* tidak dilakukan pada Unity. Sementara *GameObjects* adalah *kontainer* untuk semua Komponen lainnya. Semua objek dalam permainan disebut *game objects*. Material digunakan dan dihubungkan dengan *mesh* atau *renderer* partikel yang melekat pada *game object*. Material berhubungan dengan penyaji *Mesh* atau partikel yang melekat pada *GameObject* tersebut. Mereka memainkan bagian penting dalam mendefinisikan bagaimana objek ditampilkan. *Mesh* atau partikel Tidak dapat ditampilkan Tanpa material karena material meliputi referensi untuk *Shader* yang

digunakan untuk membuat *Mesh* atau Partikel. Material digunakan untuk menempatkan Tekstur ke *GameObjects* [7].

II.10 *Game Crazy Ball*

Game Crazy Ball ini adalah permainan sederhana yang menggunakan bola sebagai pergerakannya. Tujuan permainan ini adalah pengguna diharuskan untuk keluar dari tingkatan satu ke tingkatan berikutnya dengan menggunakan bola sebagai pergerakan *user* nya. Pengguna diharuskan menyelesaikan setiap *level* dengan cara mencapai pintu keluar untuk menyelesaikan permainan, Permainan akan selesai jika pengguna menyelesaikan semua *level*. Permainan *Crazy Ball* ini sangat unik dibandingkan *game* lainnya karena permainan ini memberikan interaksi yang unik dalam setiap *level* nya. Tantangan dalam permainan ini berupa tantangan *geometris*, musuh, benda-benda interaktif atau non interaktif yang memiliki beberapa efek fisik maupun non fisik pada karakter pemain atau geometri sekitarnya. Berikut merupakan gambar setiap *level* dalam *game Crazy Ball*.



Gambar II. 5 *Game Crazy Ball*

Game crazy ball ini akan digunakan sebagai referensi sebagai pembuatan *game ball logic* karena *game crazy ball* ini merupakan permainan yang terbilang unik dalam peraturan permainan nya. Oleh karena itu ide pembuatan *game ball logic* ini berdasarkan *game* pendahulunya yaitu *game crazy ball*, sehingga untuk cara permainan atau aturan permainan tidak jauh berbeda dengan *game crazy ball*.

1. *Game Crazy Ball*

a. Kelebihan : Pada *game* ini terdapat banyak rintangan seperti *level* 1 – *level* 10, sehingga permainan tidak mudah bosan karena setiap *level nya* menampilkan rintangan yang berbeda beda.
b. Kekurangan : Permainan ini cenderung terlalu sulit ketika berada pada *level* 4 ke atas, karena pergerakan bola yang terlalu cepat ketika digerkaan dan rintangan yang semakin sulit untuk dilewati.

2. *Game Ball Logic*
   a. Kelebihan : Pada *game* ini permaian dibuat tidak terlalu sulit dalam hal rintangan maupun pregerakan bola nya, karena rintangan akan dibuat mudah dan untuk kecepatan bola ketika digerakan akan sedikit diturunkan.
   b. Kekurangan : *Game* ini hanya menampilkan *level* 1 – *level* 3 dan hanya mempunya tiga tingkatan kesulitan yakni *level* 1 = *easy level* 2 = *medium* dan *level* 3 = *hard*

Kesimpulan dari perbandingan *game* diatas maka pada penelitian ini akan mengambil *game ball logic* sebagai *game* yang akan diterapkan teknologi *augmented reality* karena rintangan dalam setiap *level* pada *game* ini tidak terlalu sulit dan juga *level* yang terbilang sedikit dibanding *game* pendahulunya yaitu *game crazy ball*.

II.11 UML (*Unified Modeling Language*)

UML dalam sebuah bahasa untuk menentukan visualisasi, konstruksi, dan mendokumentasikan *artifact* dari sistem *software*, untuk memodelkan bisnis, dan sistem *non-software* lainnya. UML merupakan sistem arsitektur yang bekerja dalam OOAD (*Object Oriented Analysis and Design*) dengan satu bahasa yang konsisten untuk menentukan, visualisasi, konstruksi dan mendokumentasikan *artifact* yang terdapat dalam sistem. *Artifact* adalah potongan informasi yang digunakan atau dihasilkan dalam suatu proses rekayasa *software*. *Artifact* dapat berupa model, deskripsi atau *software*.

Unified Modeling Language (UML) adalah keluarga notasi grafis yang didukung oleh meta-model tunggal, yang membantu pendeskripsian dan *desain* sistem perangkat lunak, khususnya sistem yang dibangun menggunakan pemrograman brorientasi objek (OO). UML lahir dari penggabungan banyak

bahasa pemodelan grafis berorientasi objek yang berkembang pesat pada akhir 1980 dan awal 1990.

1. *Use Case Diagram*

Use Case adalah teknik untuk merekam persyaratan fungsional sebuah sistem. *Use Case* mendeskripsikan interaksi tipikal antara para pengguna sistem dengan sistem itu sendiri, dengan memberi sebuah narasi tentang bagaimana sistem tersebut digunakan.

2. *Activity Diagram*

Activity Diagram adalah teknik unutk menggambarkan logika prosedural, proses bisnis, dan jalur kerja. Dalam beberapa hal, *diagram* ini memainkan peran mirip sebuah *diagram* alir, tetapi perbedaan prinsip antara *diagram* ini dan notasi *diagram* alir adalah diagram ini mendukung *behavior paralel*.

3. *Class Diagram*

Class Diagram mendeskripsikan jenis-jenis objek dalam sistem dan berbagai macam hubungan statis yang terdapat di antara mereka. *Class Diagram* juga menunjukan properti dan operasi sebuah *class* dan batasan-batasan yang terdapat dalam hubungan-hubungan objek tersebut.UML menggunakan istilah fitur sebagai istilah umum yang meliputi properti dan operasi sebuah *class*.

4. *Sequence Diagram*

Interaction diagram menunjukan bagaimana kelompok-kelompok objek saling berkolaborasi dalam beberapa *behavior*. UML memiliki beberapa bentuk *interaction diagram* dan yang paling umum digunakan adalah *sequance diagram*. Sebuah *sequance diagram*, secara khusus, menjabarkan *behavior* sebuah sekenario tunggal. *Diagram* tersebut menunjukan sejumlah objek contoh dan pesan-pesan yang melewati objek-objek ini di dalam *use case*.

II.12 Pengujian

Pengujian adalah suatu proses mengevaluasi suatu sistem atau komponen dengan maksud untuk menemukan apakah itu memenuhi persyaratan yang ditentukan atau tidak. Dengan kata lain pengujian ialah mengidentifikasi kesenjangan, atau kesalahan. Dalam penelitian ini dilakukan pengujian *black box*.

Pengujian *black box* merupakan pengujian yang mengabaikan mekanisme internal sistem atau komponen dan fokus semata-mata pada *output* yang dihasilkan yang merespon *input* yang dipilih dan kondisi eksekusi. Pengujian yang dilakukan untuk mengevaluasi pemenuhan sistem atau komponen dengan kebutuhan fungsional tertentu. Metode ujicoba *black box* memfokuskan pada keperluan fungsional dari *software*. Karena itu ujicoba *black box* memungkinkan pengembang *software* untuk membuat himpunan kondisi *input* yang akan melatih seluruh syarat-syarat fungsional suatu program. Ujicoba *black box* bukan merupakan alternatif dari ujicoba *whitebox*, tetapi merupakan pendekatan yang melengkapi untuk menemukan kesalahan lainnya, selain menggunakan metode *whitebox*. Ujicoba *black box* untuk menemukan kesalahan dalam beberapa kategori, diantaranya :

1. Fungsi-fungsi yang salah atau hilang
2. Kesalahan *interface*
3. Kesalahan dalam struktur data
4. Kesalahan performa
5. Kesalahan inisialisasi