BAB III
ANALISIS DAN PERANCANGAN SISTEM

III.1 Analisis Masalah

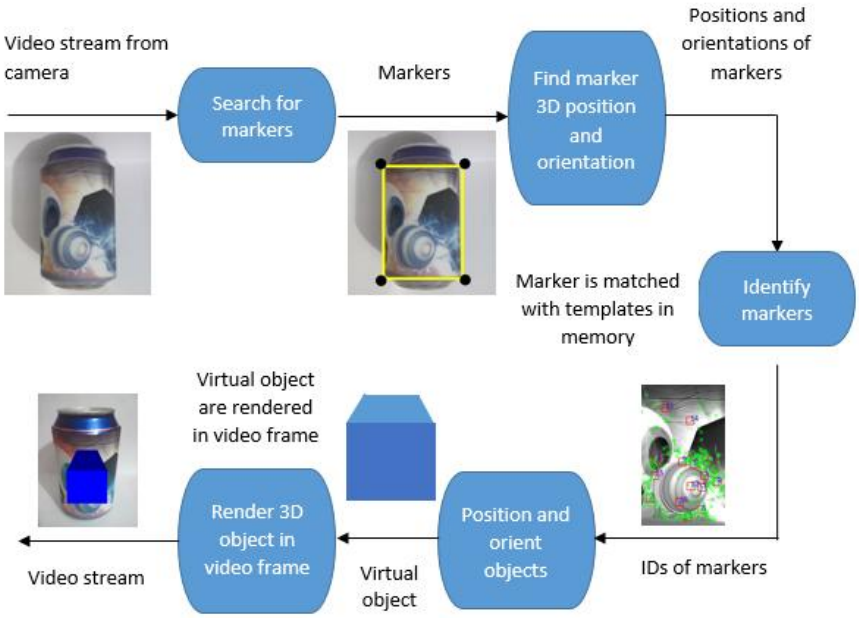
Berdasarkan dari latar belakang masalah yang sudah didapat, maka akan dijelaskan secara lebih terperinci tentang kekurangan dari *library* ARToolkit. Adapun kekurangan *library* ARtoolkit adalah dalam pelacakan dengan menggunakan metode *markerless* hanya mampu mendeteksi sebuah *marker* yang berbentuk suatu bidang datar saja, sehingga jika sebuah objek sudah dihasilkan maka objek tersebut akan terlihat cukup baik jika *marker* yang dideteksi terlihat seluruhnya pada kamera. Maka dari itu penggunaan *marker* dengan metode *markerless* pada *library* ARtoolkit kurang begitu baik jika penerapannya pada sebuah *game* karena *marker* yang dibutuhkan pada *game* haruslah *marker* yang benar benar kuat dalam pelacakannya walaupun dari berbagai sudut pandang. Dalam penelitian ini masalah yang diangkat adalah implementasi teknologi *augmented reality* pada *game ball logic* menggunakan *tracking cylindrical object* dengan tujuan melakukan pengembangan pada *library* ARToolkit yang masih terbatasnya sebuah *marker* yang akan dilacak.

III.1.1 Analisis Penyelesaian Masalah

Dalam penyelesaian masalah tersebut, maka sistem *tracking* terhadap suatu *marker* ini dibangun mengimplementasikan metode *markerless* dengan *tracking cylindrical object*. Sistem ini bekerja dengan cara melakukan *tracking* terhadap sebuah *marker* yang berbentuk silinder sehingga kamera tidak harus melakukan pelacakan *marker* secara seluruhnya tapi cukup sebagian saja *marker* yang dapat dideteksi tetapi *object* yang dihasilkan cukup baik.

III.2 Analisis Sistem

Analisis sistem menjelaskan permasalahan pada suatu sistem dan mendeskripsikan kebutuhan pada pembangunan sistem pelacakan *marker*. Untuk pelacakan *marker* menggunakan *Library* ARToolKit dengan cara kerja seperti pada gambar III.1 berikut:



Gambar III. 1 Cara Kerja ARToolKit

Pada sistem yang akan dibangun terdapat beberapa proses yang memiliki fungsi masing-masing.

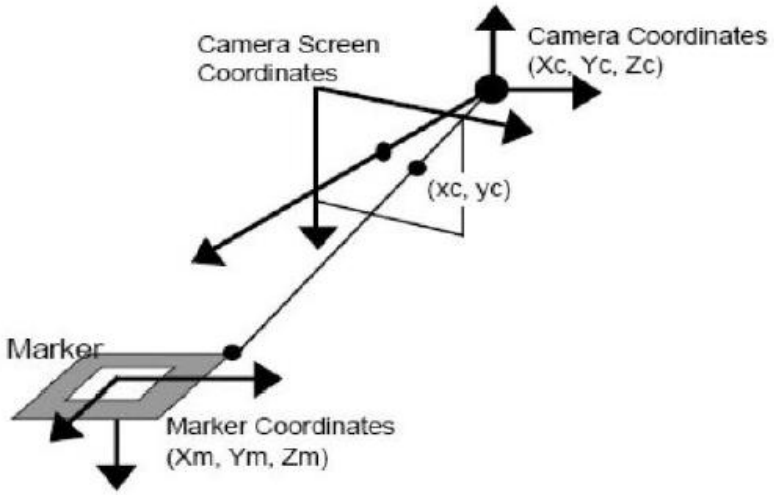
III.3.1 Analisis Sistem Yang Berlaku

Analisis ini menjelaskan tentang suatu sistem yang masih berlaku pada *library* ARToolkit, pada tahap ini juga analisis lebih difokuskan pada pelacakan *marker* yang masih berbentuk persegi empat atau bidang datar. Proses *natural feature tracking* diharuskan tertangkap kamera sepenuhnya agar objek yang dihasilkan dari *tracking* pada *marker* dapat menghasilkan objek yang cukup baik, untuk lebih jelasnya lihat seperti gambar III.2:

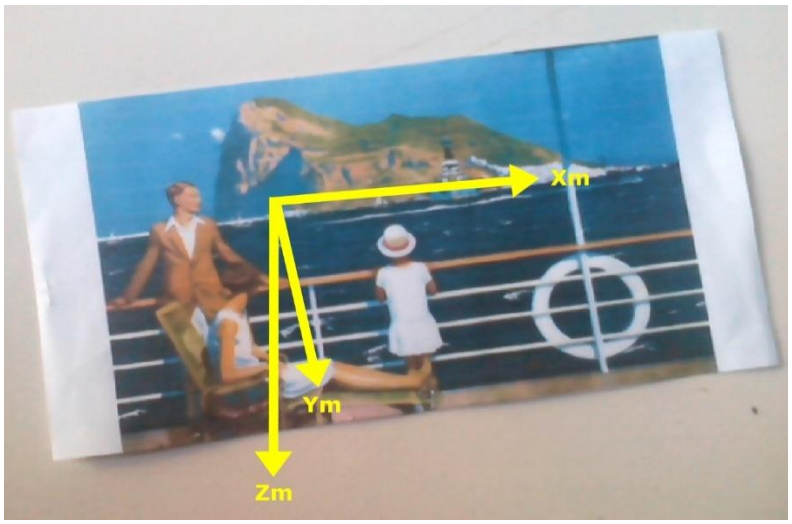


Gambar III. 2 *Marker NFT* Biasa

Sistem akan mengenali *marker* dengan menentukan titik kordinat dari *marker* dan kamera, seperti pada gambar III.3:

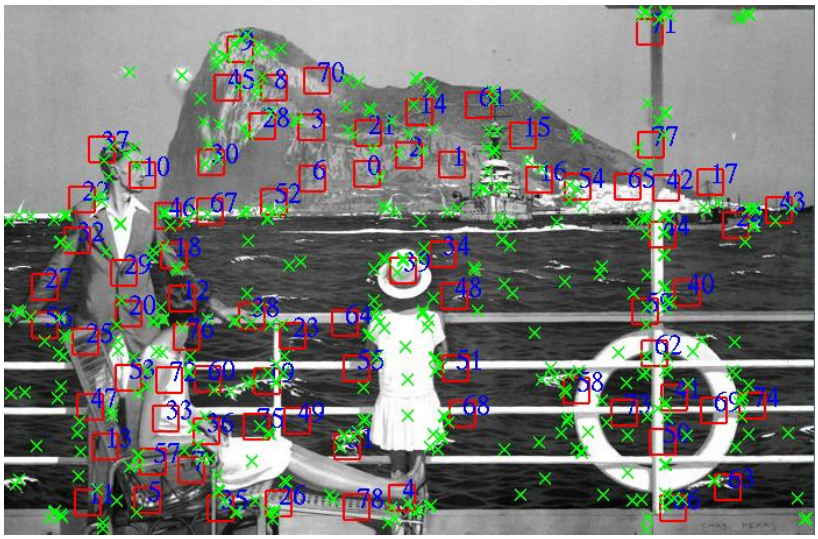


Gambar III. 3 Koordinat *Marker* dan Koordinat Kamera



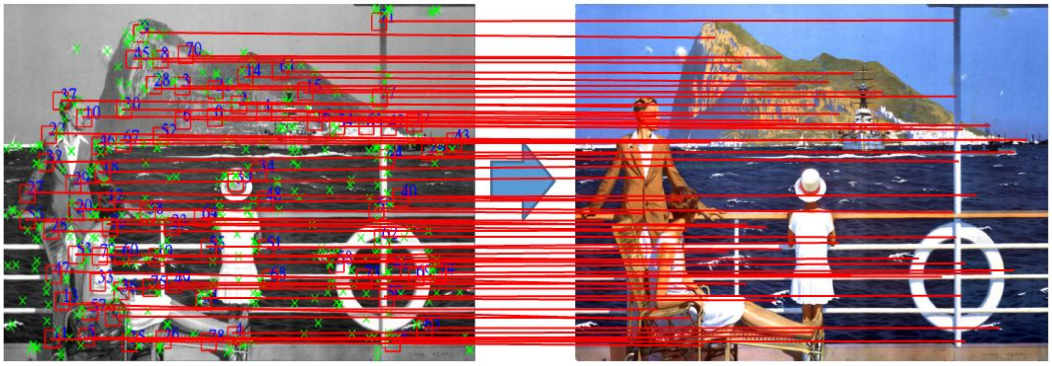
Gambar III. 4 Koordinat *Marker*

Pada gambar III.4 dapat dijelaskan bahwa kordinat *marker* akan dikenali sebagai Xm, Ym dan Zm sedangkan kordinat kamera sebagai Xc, Yc dan Zc. Setelah proses menentukan titik kordinat selesai maka selanjutnya masuk ke dalam tahap menandai *marker* dengan menentukan *feature point*, *feature point* tersebut akan berguna sebagai persamaan *marker* yang nantinya akan dibandingkan dengan *marker* yang dideteksi oleh kamera dan apabila sesuai maka objek akan ditampilkan. Seperti pada gambar III.5 berikut:



Gambar III. 5 *Feature Point*

Hijau merupakan *feature freak*, sedangkan merah mewakili sampel untuk persamaan *marker*. Gambar III.5 diatas menampilkan *feature freak* sejumlah 512 *point* dan *feature* berwarna merah yang akan digunakan sebagai persamaan *marker* sejumlah 79 *point*. Jika proses ini selesai maka selanjutnya masuk ke dalam tahap menampilkan objek *augmented reality*, dalam tahap ini dilakukanlah pencocokan *marker* dengan menggunkan *feature point* yang berwarna merah seperti gambar III.5 diatas. Lebih jelasnya dapat dilihat pada gambar III.6 berikut:



Gambar III. 6 Pencocokan *Marker*

Proses ini merupakan tahapan terakhir dari sistem karena jika *marker* sudah cocok maka akan menghasilkan sebuah objek *augmented reality* yang akan langsung ditampilkan diatas *marker*, seperti gambar III.7 berikut:



Gambar III. 7 Objek *Augmented Reality*

III.3 Analisis *Game*

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*. Pada *game ball logic* ini pula ada beberapa area permaianan / *map* yang diambil secara langsung dari *game crazy ball*. Di dalam analisis *game* ini akan dijelaskan mengenai gambaran *game crazy ball* dan *game ball logic*, analisis dilakukan dengan maksud untuk mengidentifikasi dan mengevaluasi kebutuhan – kebutuhan yang diharapkan sehingga dapat diusulkan perbaikan – perbaikannya. Analisis yang akan dibahas mengenai analisis *storyline*, *gameplay*, jumlah *level* dan teknologi *augmented reality*. Untuk teknologi *augmented reality* hanya akan dibahas pada *game ball logic.*

III.3.1 *Game Crazy Ball*

Pada *game crazy ball* analisis yang akan dibahas mengenai *storyline*, *gameplay* dan jumlah *level*. Untuk teknologi *augmented reality* tidak diterapkan pada *game* ini karena *crazy ball* hanya sebuah *game* yang akan digunakan sebagai ide untuk pembuatan *game ball logic* yang akan diterapkan teknologi *augmented reality*.

III.3.1.1 *Storyline*

Game crazy ball menceritakan tentang sebuah bola berwarna merah yang diharuskan untuk keluar dari sebuah tempat untuk mengakhiri permainan nya, tempat itu biasa disebut dengan *level*. Akan tetapi untuk keluar dari tempat / *level* tersebut bola merah dihadapkan dengan rintangan rintangan yang ada didepan matanya, seperti sebuah dinding, teka-teki sebuah jalan, objek yang bergerak dan lain-lain.

III.3.1.2 *Gameplay*

Gameplay merupakan interaksi antara *user* dengan suatu *game* yang akan dimainkannya yang berkenaan dengan aturan bermain dari *game* tersebut. Berikut merupakan *gameplay* pada *game crazy ball*:

1. *User* sudah langsung dihadapkan dengan *menu* berupa *level* 1 – *level* 10.
2. *User* tinggal memilih *level* yang diinginkan untuk memainkan *game* ini.
3. Jika *user* akan memilih *level*, maka *user* diharuskan menggerakan bola.
4. Untuk pergerakan bola *user* hanya tinggal menggunakan arah panah.
5. Ketika bola bersentuhan dengan objek *level* yang akan di pilih, misalnya *level* 1. Secara otomatis *game* akan menampilkan *level* 1.
6. Di dalam *level* 1 ini lah permainan yang sebenarnya di mulai, karena *user* diharuskan mencari jalan untuk keluar dari area permaianan *level* 1.

III.3.1.3 Jumlah *level*

Untuk jumlah *level* pada *game crazy ball* ada 10 *level*, tentunya dari masing-masing *level* berbeda tingkat kesulitan nya. Pada masing-masing *level game* akan disuguhkan dengan berbagai macam rintangan yang akan menghadang pergerakan bola merah untuk mencapai tombol keluar / *exit*.

III.3.2 *Game Ball Logic*

Game ball logic ini memiliki kelebihan jika diterapkan teknologi *augmented reality* seperti ketika *user* memainkan *game* akan terasa lebih interaktif lagi karena *user* dapat mengendalikan penuh pergerakan bolanya dengan sebuah objek silinder yang nantinya akan dideteksi oleh *webcam*, *game* ini akan mengambil latar pada lingkungan nyata kemudian digabungkan dengan objek – objek 3D yang ditampilkan pada layar komputer melalui *webcam*.

III.3.2.1 *Storyline*

Game ball logic ini menceritakan mengenai bola besi yang diharuskan untuk mencari jalan keluar dari sebuah arena / *map* dengan rintangan nya berupa objek berbentuk 3D yang disusun sedemikian rupa untuk menghalangi bola besi ke arah tempat tujuan. Tempat tujuan yang dimaksud akan mengarahkan bola besi keluar dari arena / *map*.

III.3.2.2 *Gameplay*

Gameplay merupakan interaksi antara *user* dengan suatu *game* yang akan dimainkannya yang berkenaan dengan aturan bermain dari *game* tersebut. Berikut merupakan *gameplay* pada *game ball logic*:

1. Pemain memilih *menu start* untuk memainkan *game* ini.
2. Pemain mengarahkan *marker* kepada *webcam* untuk dideteksi.
3. Permainan pada *game* akan langsung dimulai ketika *marker* berhasil dideteksi.
4. Pemain memulai menggerakan *marker* ke arah kiri, kanan atau atas, bawah untuk menggerakan bola.
5. Pada tahap ini pemain diharuskan memilih *level* dengan cara bola digerakan menuju target *level* yang diinginkan.
6. Jika pemain sudah memilih *level*, misalnya *level* 1. Maka pemain akan langsung berpindah menuju area permainan *level* 1.
7. Di dalam *level* 1 ini lah permainan yang sebenarnya di mulai, karena *user* diharuskan mencari jalan untuk keluar dari area permaianan *level* 1.

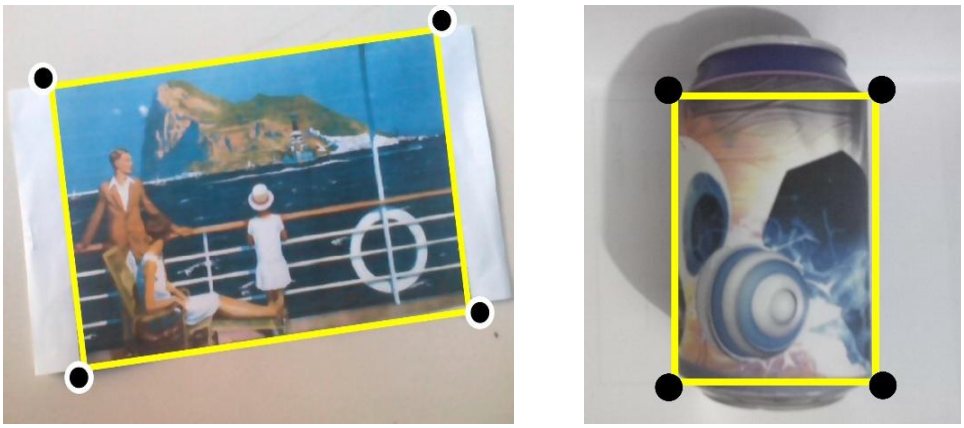
III.3.2.3 Jumlah *Level*

Untuk jumlah *level* pada *game ball logic* ini ada 3 *level* yang tentunya memiliki tingkatan kesulitan yang berbeda beda, seperti *level* 1. Pada *level* ini tingkatan nya dinyatakan *easy, level* 2 tingkatan nya dinyatakan *medium* dan *level* 3 tingkatan nya dinyatakan *hard*.

III.3.2.4 Analisis Teknologi *Augmented Reality*

Augmented reality pada *game ball logic* ini digunakan sebagai media interaksi antara pemain dengan *game*. *Library augmented reality* yang digunakan dalam *game* ini adalah ARToolkit. Berikut merupakan tahap yang menjelaskan tentang bagaimana mengubah *marker* berbentuk persegi empat menjadi berbentuk

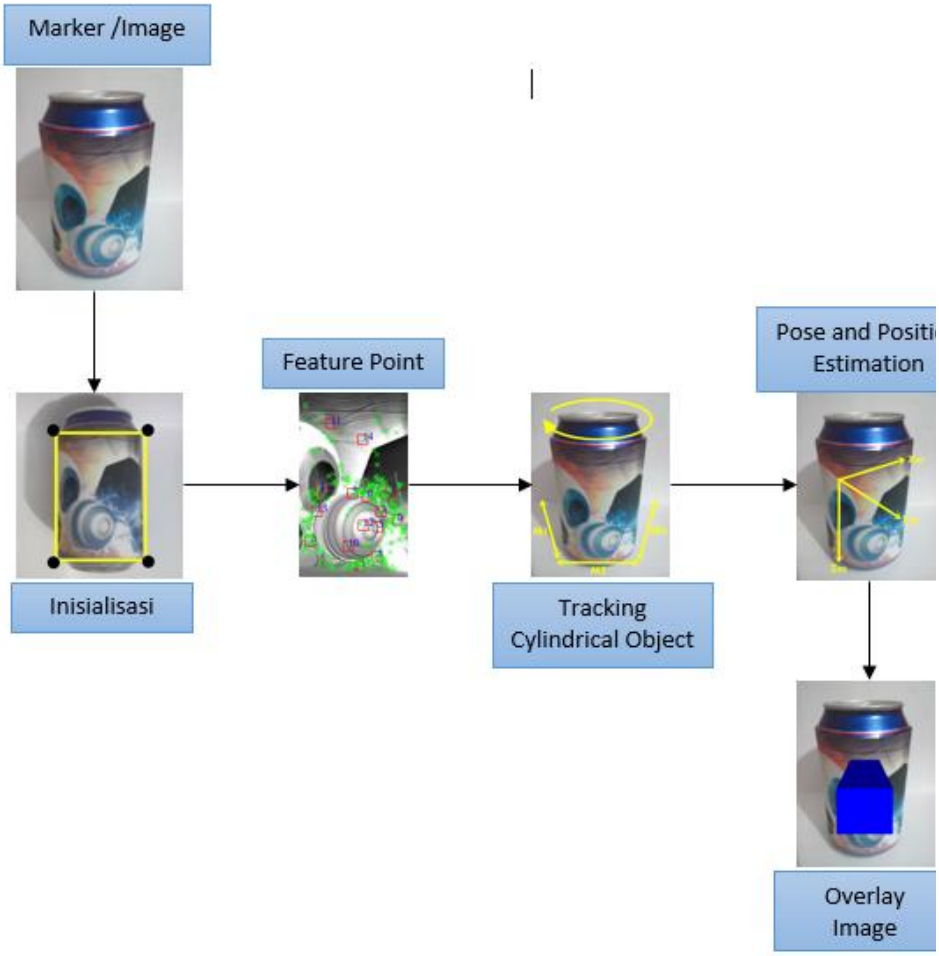
silinder. Untuk lebih jelasnya perbedaan antara proses *Simple NFT* dengan proses *Tracking Cylindrical Object* dapat dilihat pada gambar III.8:



(a) Proses *Simple NFT* (b) Proses *Tracking Cylindrical Object*

Gambar III. 8 *Simple NFT* dan *Tracking Cylindrical Object*

Pada gambar III.8 dapat dijelaskan bahwa proses *simple* NFT (*Natural Feature Tracking*) menangkap *marker* secara menyeluruh karena dukungan dari suatu bidang datar, sedangkan *tracking Cylindrical Object* menangkap *marker* hanya sebagian saja dikarenakan *marker* ini bidangnya tidak datar melainkan berbentuk silinder sehingga dapat dipastikan *marker* yang akan nampak pada kamera hanya sebagian saja. Alur proses pembuatan untuk sistem pelacakan *marker* dapat dilihat pada gambar III.9:



Gambar III. 9 Alur Proses Pembuatan

Marker atau *image* ini akan digunakan sebagai kontrol pergerakan bola pada permainan *ball logic*, sehingga *marker* ini selanjutnya akan masuk ke dalam tahap inisialisasi, *feature point*, *tracking cylindrical object*, *pose and position estimation*, *overlay image*. Untuk lebih jelasnya berikut adalah rincian dari data masukan dan masing-masing proses.

1. *Marker/Image*

   Marker/Image adalah suatu penanda/*marker* yang berupa *image*, dimana *marker* tersebut nantinya akan dilakukan pelacakan menggunakan metode *markerless* (*natural features tracking*).

2. Inisialisasi

Inisialisasi adalah masukan berupa gambar, pada tahap ini *image* akan dikenali terlebih dahulu sebagai sebuah *marker* yang nantinya akan dilatih sebagai *markerless* (*natural Feature Tracking*). Untuk proses bagaimana *image* tersebut dapat dikenali sebagai *marker* maka diperlukan sebuah *tools* yang sebenarnya sudah tersedia dalam *library* ARToolkit yaitu genTexData, berikut tahap-tahap nya.

1) Buka folder“bin”.
2) Buka Command Prompt window, dan ketik“cd ” (dengan spasi).
3) Pilih path bar (ditandai), dan *copy* (ctrl+c) teks itu.
4) Paste teks itu(ctrl+v) setelah kata“cd ”pada command prompt.
5) Tekan “Enter” dalam Command Prompt window.

Setelah kelima tahap tersebut diikuti maka tahap selanjutnya tinggal mengubah *image* menjadi *marker* dengan memanggil *tools* genTexData dan memasukan *file image* yang akan diubah menjadi *marker*.

```
D:\KULIAH\SKRIPSI\ARTOOLKIT\bin>genTexData marker\cylinder\M2.JPG
```

3. Feature Point

Fungsi dari proses ini adalah untuk menandai *image* yang nantinya akan digunakan sebagai pencocokan *marker*, untuk pembuatan *feature point* menggunakan *tools* yang sudah tersedia di *library* ARToolkit yaitu dispFeatureSet. Proses nya tidak jauh berbeda dengan penggunaan *tools* sebelum nya yaitu genTexData, hanya saja dalam proses ini tidak memanggil *file* gambar dengan format JPG lagi. Tapi dalam proses ini memanggil *file* gambar yang sudah dilatih dengan menggunakan *tools* genTexData. Berikut merupakan contoh tahapan nya.

```
D:\KULIAH\SKRIPSI\ARTOOLKIT\bin>dispFeatureSet marker\cylinder\M2.iset
```

Setelah tahap tersebut dilaksanakan maka akan tampil berupa informasi *feature point*, seperti berikut

```
fset: Num of feature point: 20
fset3: Num of feature point : 606
```

Fset merupakan fitur point berwarna merah yang nantinya akan digunakan untuk pencocokan *marker*, sedangkan fset3 merupakan *feature freak*.

4. Proses *Tracking Cylindrical Object*

Proses ini menjelaskan tentang bagaimana mengubah *marker* yang berbentuk persegi empat menjadi berbentuk silinder. Pada proses ini *marker* hanya dapat terlihat sebagian dikamera karena *merker* berbentuk bidang silinder, seperti pada gambar III.10:



Gambar III. 10 *Tracking Cylindrical Object*

Untuk menampilkan objek *augmented reality* dengan bidang silinder ini diharuskan dapat membaca *marker* walaupun sebagian sisi *marker* lainnya tidak terlihat dikamera. Maka dari itu cara yang dilakukan agar sisi dari *marker* yang tidak terlihat dikamera dapat dikenali dengan memberikan tambahan inisialisasi *marker* dalam sistem dan memberikan nilai transform yang dapat menyesuaikan dengan kamera, untuk lebih jelasnya dapat dilihat pada gambar III.11:



Gambar III. 11 Inisialisasi *Marker* dan Pemberian Nilai *Transform*

Dalam gambar III.11 dapat dijelaskan bahwa untuk mengenali *marker* yang berada pada sisi yang tidak terlihat oleh kamera solusinya dengan memberikan

tambahan inisialisasi *marker* dan penyesuaian nilai *transform* berdasarkan kamera, seperti:

M1 = *Marker* 1

M2 = *Marker* 2

M3 = *Marker* 3

Untuk nilai *transform* didapat dari kalibrasi kamera, sedangkan kalibrasi kamera dibagi menjadi dua bagian yaitu kalibrasi parameter *internal* dan kalibrasi parameter *eksternal*. Untuk kalibrasi parameter *eksternal* menghasilkan matriks ekstrinsik yang merupakan orientasi posisi dari kamera terhadap *world coordinate*. Matriks ekstrinsik ini terdiri atas matriks *rotasi* (R) dan matriks *translasi* (T).

$$R = \begin{bmatrix} r11 & r12 & r13 \\ r21 & r22 & r23 \\ r31 & r32 & r33 \end{bmatrix}$$

Pengaturan Posisi

Pengaturan Rotasi

Pengaturan Skala

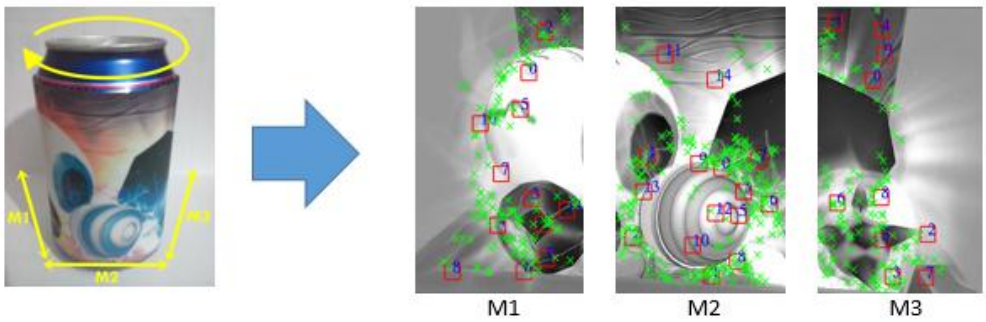
Untuk baris ke satu yang ditunjukan oleh r11, r12 dan r13 adalah untuk pengaturan posisi yang memiliki nilai tetap, contohnya disini untuk penetuan nilai nya diisi dengan nilai r11 = 0, r12 = 0, r13 = 0. Nilai tersebut bersifat tetap dan tidak akan berubah – ubah. Pada baris selanjutnya dan sampai baris terakhir yang diisi oleh nilai rotasi dan skala nilainya juga sama bersifat tetap dan tidak akan berubah – ubah.

Setelah proses inisialisasi *marker* dan pemberian nilai *transform* selesai maka tahap selanjutnya yaitu pengenalan titik kordinat pada suatu *marker*, lebih jelasnya seperti gambar III.12:



Gambar III. 12 Koordinat *Marker* TCO

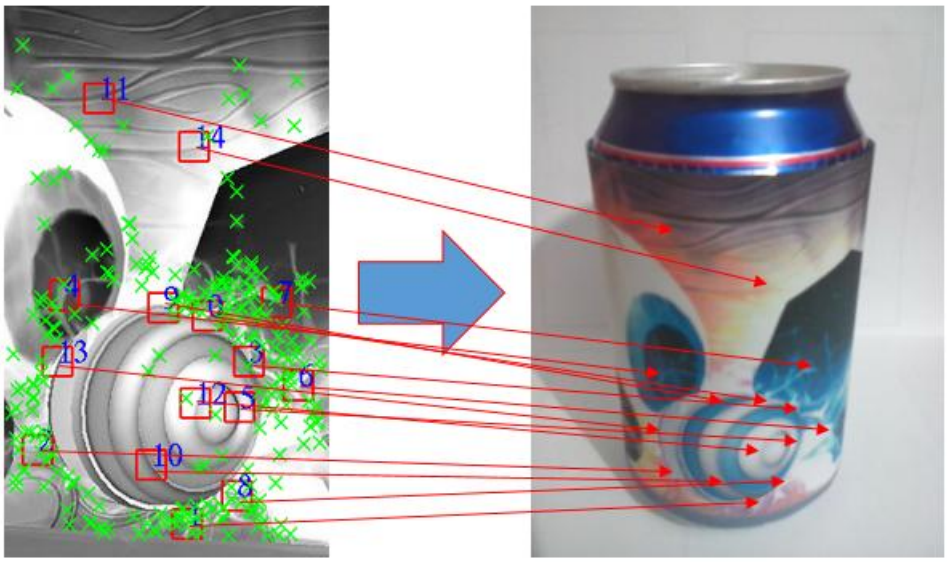
Pada gambar diatas dapat dijelaskan bahwa koordinat *marker* akan dikenali sebagai Xm, Ym dan Zm. Setelah proses menentukan titik kordinat selesai maka selanjutnya masuk ke dalam tahap menandai *marker* dengan menentukan *feature point*, *feature point* tersebut akan berguna sebagai persamaan *marker* yang nantinya akan dibandingkan dengan *marker* yang dideteksi oleh kamera dan apabila sesuai maka objek akan ditampilkan. Seperti pada gambar III.13:



Gambar III. 13 *Feature Point* TCO

Dalam gambar III.13 dapat dijelaskan bahwa M1, M2 dan M3 dilakukan proses *feature point* nya secara terpisah. Jika proses ini selesai maka selanjutnya masuk ke dalam tahap menampilkan objek *augmented reality*, dalam tahap ini dilakukanlah pencocokan *marker* dengan menggunkan *feature point* yang berwarna

merah seperti gambar III.13 diatas. Lebih jelasnya dapat dilihat pada gambar III.14 berikut:



Gambar III. 14 Pencocokan *Marker* TCO

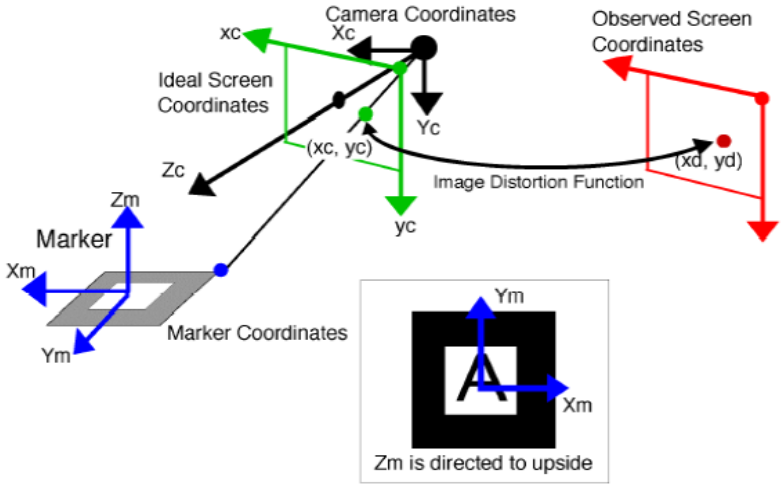
Proses ini merupakan tahapan terakhir dari sistem karena jika *marker* sudah cocok maka akan menghasilkan sebuah objek *augmented reality* yang akan langsung ditampilkan diatas *marker*, seperti gambar III.15:



Gambar III. 15 Objek *Augmented Reality* TCO

5. *Pose and Position Estimation*

Pada tahap ini juga akan dilakukan *pattern normalization* dan *template matching*, dalam proses ini sudut dari lensa kamera tidak tegak lurus terhadap *marker* ketika mengambil *video*, sudut-sudut *marker* yang dibentuk oleh sisi-sisi segi empat tidak 90°. Hal ini membuat pola yang ada didalam *marker* tidak dapat dikenali dengan baik. *Pattern normalization* berperan untuk mengubah sudut *marker* yang tidak 90° menjadi 90° agar pola dapat dikenali dan dicocokkan menggunakan *template matching* dengan pola (*template*) yang telah ada pada sistem untuk memperoleh positif ID dari *marker* tersebut. Sebuah gambar, foto, maupun nama dapat dijadikan pola pada sebuah *marker* agar sistem dapat mengenali pola itu. Untuk menaruh objek 3D tepat diatas *marker*, sistem perlu mengetahui koordinat dari *marker* dan kamera.



Gambar III. 16 Koordinat *Marker* dengan Koordinat Kamera

6. *Overlay Image*

Langkah terakhir yang dilakukan adalah menggambar objek *virtual* 3D pada *frame video* tepat diatas permukaan *marker* dan hasilnya dapat dilihat pada keluaran *video* nya. Dengan demikian model objek seolah olah ada diatas *marker*.

Ball logic adalah *game* 3d sederhana, bila dibandingkan dengan *game* pendahulunya yaitu *crazy ball*. *Game ball logic* ini lebih mudah dalam hal rintangan nya karena *game* ini akan diterapkan teknologi *augmented reality* dalam hal pergerakan bola oleh *user* sehingga dibutuhkan rintangan yg tidak terlalu rumit agar

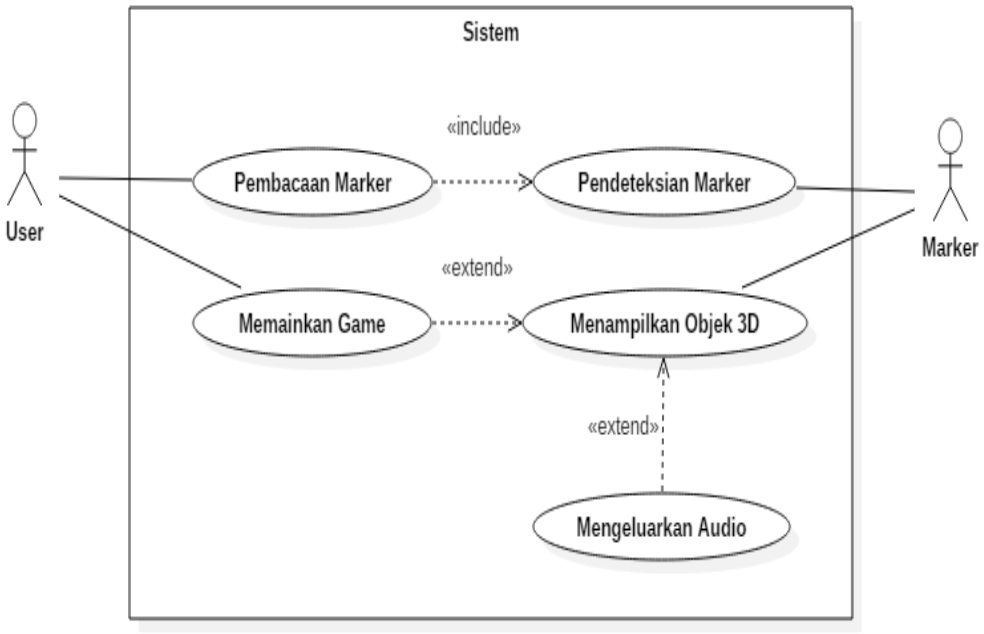
pengguna dapat memainkan dengan lebih mudah dan dalam jumlah level *game* ini lebih sedikit dibanding *crazy ball* yaitu sekitar 3 *level* saja yang meliputi tingkatan *level* 1 = *easy*, *level* 2 = *medium* dan *level* 3 = *hard*.

III.4 Analisis Kebutuhan Fungsional

Analisis kebutuhan fungsional dilakukan untuk memberikan gambaran dari data yang ada pada aplikasi yang akan dibangun. Adapun gambaran aliran data tersebut dimodelkan dalam *Use Case diagram*, *Activity diagram*, *Sequence diagram,* dan *Class diagram*. Kebutuhan fungsional dalam pembangunan aplikasi ini dimodelkan dengan menggunakan UML (*Unified Modeling Language*). Tahapan pemodelan dalam analisis ini adalah sebagai berikut:

III.4.1 *Use Case Diagram*

Use case diagram memperlihatkan hubungan-hubungan yang terjadi antara aktor dengan *use case* dalam sistem. Pada sistem ini terdapat dua aktor yaitu *user* dan *marker*. Untuk diagram *use case* pada sistem ini dapat dilihat pada gambar III.21:



Gambar III. 17 *Use Case Diagram*

1. Identifikasi Aktor

Berikut adalah identifikasi aktor yang ada pada *use case diagram* sebagai berikut:

Tabel III. 1 Identifikasi Aktor

No	Aktor	Deskripsi
1	*User*	Aktor dapat mengakses dan menggunakan fitur-fitur yang tersedia pada aplikasi
2	*Marker*	Merupakan *inputan* untuk proses *augmented reality*

2. Identifikasi *Use Case Diagram*

Berikut adalah Identifikasi dari setiap *use case* yang ada pada *use case diagram*:

Tabel III. 2 Identifikasi *Use Case Diagram*

No	*Use Case*	Deskripsi
1	Pembacaan *marker*	Fungsionalitas untuk membaca *marker*
2	Pendeteksian *Marker*	Fungsionalitas untuk mengenali *marker*
3	Menampilkan Objek 3D	Fungsionalitas untuk menampilkan objek 3D
4	Mengeluarkan *Audio*	Fungsionalitas untuk mengeluarkan *audio*
5	Memainkan *Game*	Fungsionalitas untuk memainkan *game*

III.4.2 Skenario *Use Case Diagram*

Pada skenario *use case diagram* menjelaskan urutan langkah-langkah dalam sistem peringkasan *teks*, berikut adalah skenario *use case diagram* pada aplikasi:

1. Skenario *Use Case* Pembacaan *Marker*

Skenario *use case* pembacaan *marker* dapat dilihat pada tabel 3.

Tabel III. 3 Skenario *Use Case* Pembacaan *Marker*

Use Case Name	Pembacaan *Marker*
Goal In Context	Aplikasi dapat menampilkan objek 3D dan *audio*

Precondition	*Marker* belum terdeteksi	
Description	Fungsionalitas untuk melakukan pembacaan terhadap *marker*	
Related Use Case	-	
Seccessful End Condition	Aplikasi dapat mengenali *marker* dan menampilkan objek 3D beserta *audio*	
Failed End Condition	Aplikasi tidak dapat mengenali *marker*	
First Actors	*User*	
Second Actors	*Marker*	
Include	Pendeteksian *Marker*	
Extend	-	
Trigger	Pengguna memilih tombol *start* pada *menu*	
Main Flow	*Step*	*Actions*
	1	*User* memilih tombol *start* pada *menu*
	2	Pengguna mengarahkan *marker* pada kamera
	3 *Include* : pendeteksian *marker*	Aplikasi mendeteksi *marker*
	4	Aplikasi merender objek 3D
	5	Aplikasi menampilan objek 3D sesuai *marker* yang terdeteksi
Extensions	*Step*	*Actions*
	4.1	Aplikasi tidak dapat mengenali *marker* yang terdetesi

	5.1	Aplikasi tidak dapat merender objek 3D sesuai dengan *marker*

2. Skenario *Use Case* Pendeteksian *Marker*

Skenario *use case* pendeteksian *marker* dapat dilihat pada tabel 4.

Tabel III. 4 Skenario *Use Case* Pendeteksian *Marker*

Use Case Name	Pendeteksian *Marker*	
Goal In Context	Aplikasi dapat mengenali *marker* yang terdeteksi	
Precondition	*Marker* belum terdeteksi	
Description	Fungsionalitas untuk pendeteksian *marker*	
Related Use Case	-	
Seccessful End Condition	Aplikasi dapat mengenali *marker*	
Failed End Condition	Aplikasi tidak dapat mengenali *marker*	
First Actors	*User*	
Second Actors	*Marker*	
Extend	Menampilkan objek 3D	
Trigger	Kamera dapat mendeteksi *marker*	
Main Flow	*Step*	*Actions*
	1	*User* mengarahkan *marker* pada kamera
	2	Aplikasi menginisialisasi *marker*
	3	Aplikasi mengidentifikasi *marker*
	4	Aplikasi dapat melakukan proses *grayscale*
	5	Aplikasi dapat melakukan proses pendeteksian fitur
	6	Aplikasi mampu mendeteksi *marker* sesuai dengan *library*
Extensions	*Step*	*Branching Actions*

	3.1	Aplikasi tidak mengidentifikasi *marker*
	6.1	Aplikasi tidak dapat mendeteksi *marker*

3. Skenario *Use Case* Menampilkan Objek 3D

Skenario *use case* menampilkan objek 3d dapat dilihat pada tabel 5.

Tabel III. 5 Skenario *Use Case* Menampilkan Objek 3D

Use Case Name	Menampilkan Objek 3D	
Related Requirements	-	
Goals In Context	Aplikasi menampilkan objek 3D	
Preconditions	*Marker* yang terdeteksi belum dapat dikenali	
Seccessful End Condition	Aplikasi menampilkan objek 3D sesuai *marker* yang dideteksi tepat diatas *marker*	
Failed End Condition	Aplikasi tidak dapat menampilkan objek 3D	
First Actors	*User*	
Second Actors	*Marker*	
Extend	Mengeluarkan *audio*, mengambil tekstur *marker*, memainkan *game*	
Trigger	*User* menjalankan permainan	
Main Flow	*Step*	*Actions*
	1	Pengguna mengarahkan *marker* pada kamera
	2	Kamera membaca *marker*
	3	Aplikasi mengenali *marker*
	4	Aplikasi merender objek 3D
	5	Objek 3D ditampilkan tepat diatas *marker*
Extensions	*Step*	*Actions*
	3.1	Aplikasi tidak dapat menemukan *marker*

		Aplikasi tidak merender objek 3D
		Aplikasi tidak menampilkan objek 3D

4. Skenario *Use Case* Mengeluarkan *Audio*

   Skenario *use case* mengeluarkan *audio* dapat dilihat pada tabel 7.

Tabel III. 6 Skenario *Use Case* Mengeluarkan *Audio*

Use Case Name	Mengeluarkan *Audio*	
Related Requirements	-	
Goals In Context	Aplikasi mengeluarkan *audio* dari objek ditampilkan	
Preconditions	Objek 3D sudah ditampilkan	
Seccessful End Condition	Aplikasi dapat mengeluarkan *audio* dari objek 3D yang ditampilkan	
Failed End Condition	Akan aplikasi tidak dapat mengeluarkan *audio*	
First Actors	*User*	
Second Actors	*Marker*	
Trigger	Objek 3D telah ditampilkan	
Main Flow	*Step*	*Actions*
	1	User menekan tombol suara objek
	2	Aplikasi mengeluarkan *audio* objek
Extensions	*Step*	*Actions*
	2.1	Aplikasi tidak dapat mengeluarkan *audio*

5. Skenario *Use Case* Memainkan *Game*

   Skenario *use case* memainkan *game* dapat dilihat pada tabel 6.

Tabel III. 7 Skenario *Use Case* Memainkan *Game*

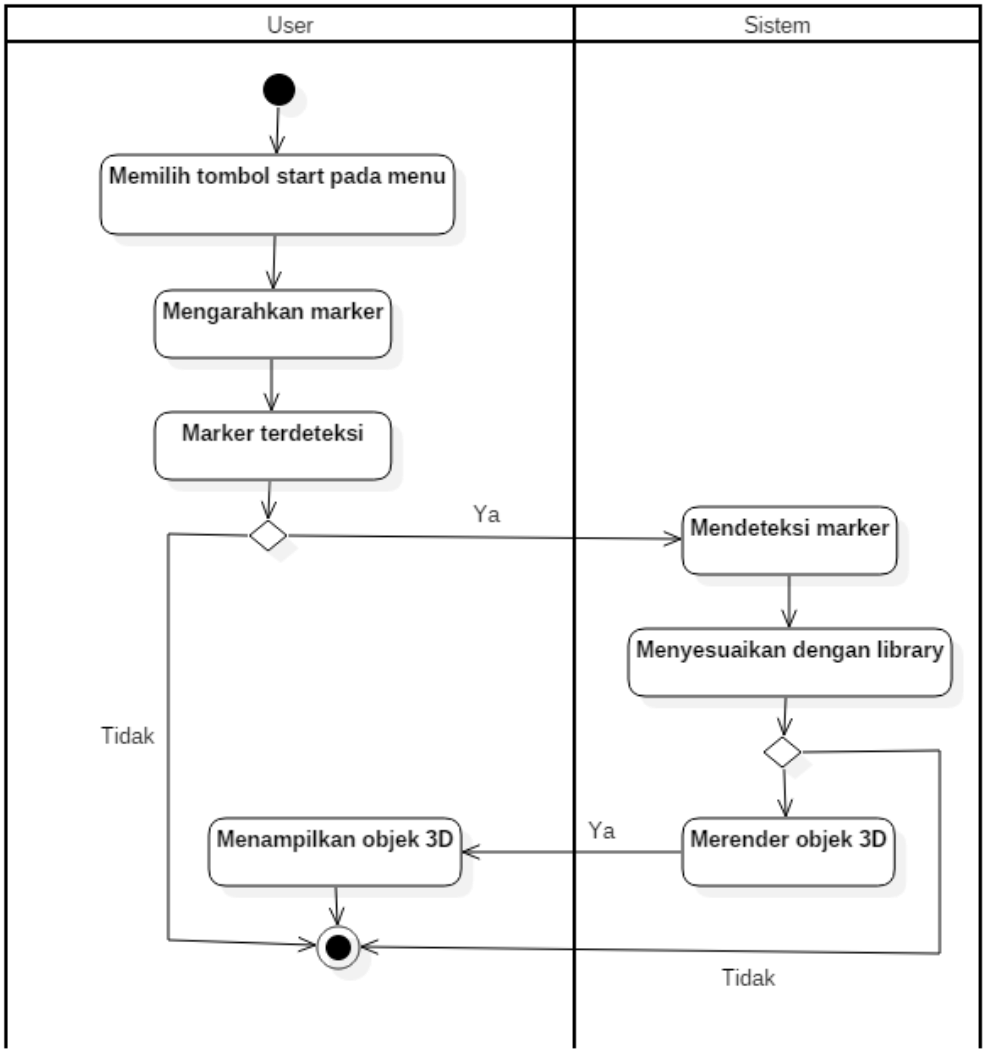
Use Case Name	Memainkan *Game*	
Related Requirements	-	
Goals in Context	Memainkan *game* setelah *marker* silinder terdeteksi	
Preconditions	Aplikasi berhasil mendeteksi *marker* silinder	
Seccessful End Condition	Aplikasi dapat dimainkan	
Failed End Condition	Aplikasi tidak dapat dimainkan	
First Actors	Sistem	
Second Actors	*User*	
Trigger	Objek 3D telah ditampilkan	
Main Flow	*Step*	*Actions*
	1	Aplikasi menampilkan objek 3D
	2	User menggerakan *marker* sehingga objek 3D bergerak
	3	Bola akan bergerak sesuai pergerakan objek 3D
Extensions	*Step*	*Branching Actions*
	3.1	Bola tidak dapat bergerak mengikuti pergerakan objek 3D

III.4.3 *Activity Diagram*

Activity diagram pada penelitian ini menjelaskan tentang alur kerja tahapan-tahapan aktivitas dari setiap *use case*. Berikut adalah masing-masing *activity diagram:*

1. *Activity diagram* pembacaan *marker*

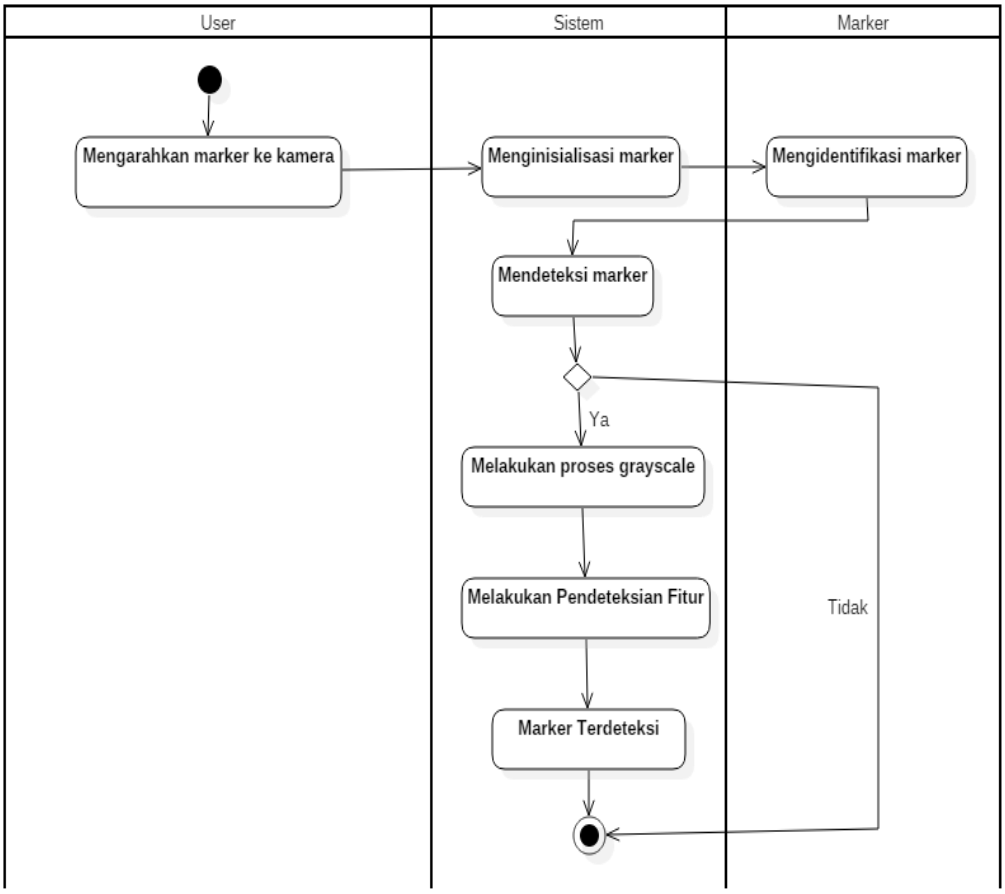
Activity diagram pembacaan *marker* menjelaskan tentang aktivitas cara melakukan pembacaan terhadap *marker* yang berbentuk silinder ke dalam sistem. *Activity diagram* pembacaan *marker* dapat dilihat pada gambar III.19.



Gambar III. 18 *Aktivity Diagram* Pembacaan *Marker*

2. *Activity diagram* pendeteksian *marker*

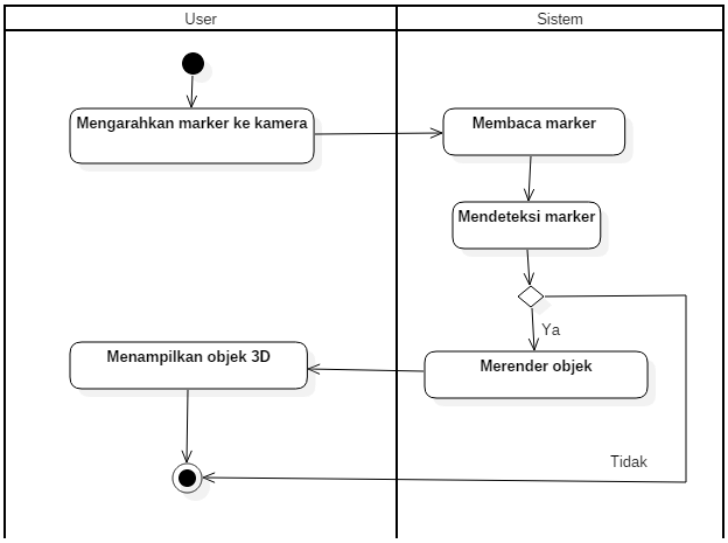
Activity diagram pendeteksian *marker* menjelaskan tentang aktivitas cara pendeteksian terhadap *marker* di dalam sistem. *Activity diagram* pendeteksian *marker* dapat dilihat pada gambar III.20.



Gambar III. 19 *Activity Diagram* Pendeteksian *Marker*

3. *Activity diagram* menampilkan *objek 3d*

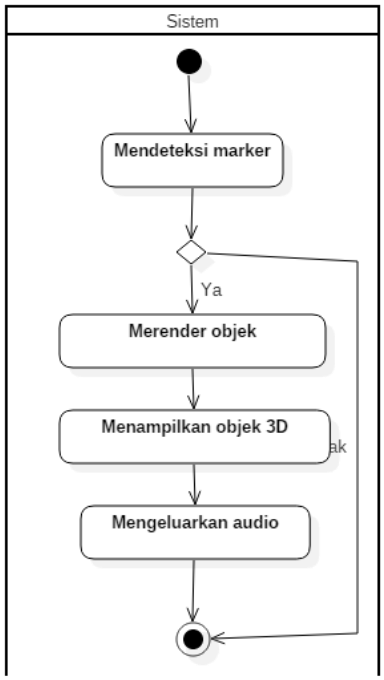
Activity diagram menampilkan *objek 3d* ini menjelaskan tentang aktivitas yang terjadi ketika akan menampilkan *objek 3d*. *Activity diagram* tersebut dapat dilihat pada gambar III.20.



Gambar III. 20 *Activity Diagram* Menampilkan *Objek 3D*

4. *Activity diagram* mengeluarkan *audio*

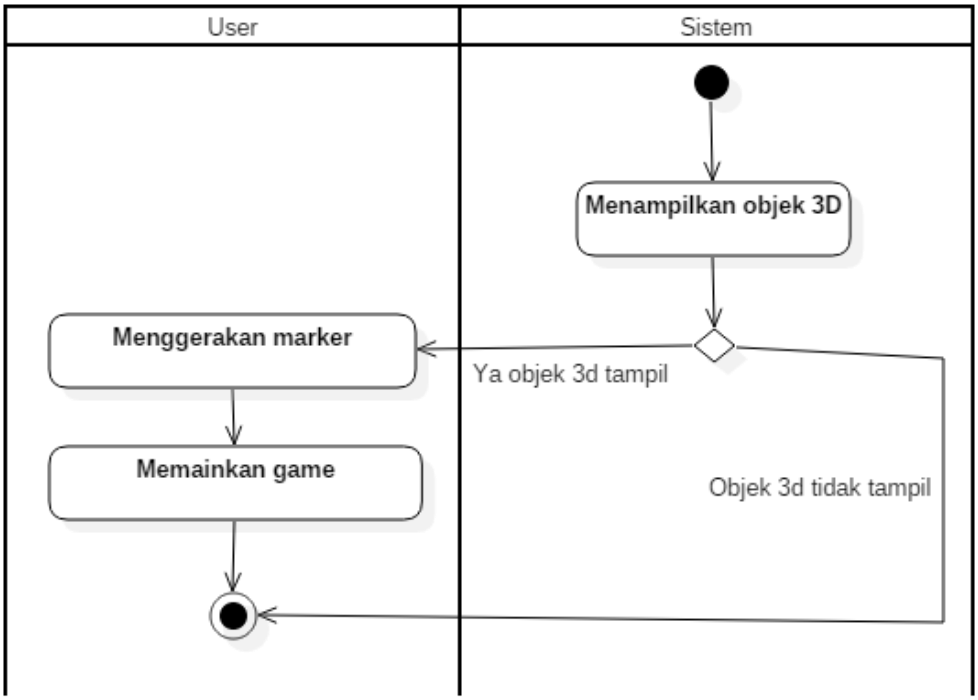
Activity diagram mengeluarkan *audio* menjelaskan tentang aktivitas cara mengeluarkan *audio* setelah *marker* terdeteksi oleh sistem. *Activity diagram* mengeluarkan *audio* dapat dilihat pada gambar III.21.



Gambar III. 21 *Activity Diagram* Mengeluarkan *Audio*

5. *Activity diagram* memainkan *game*

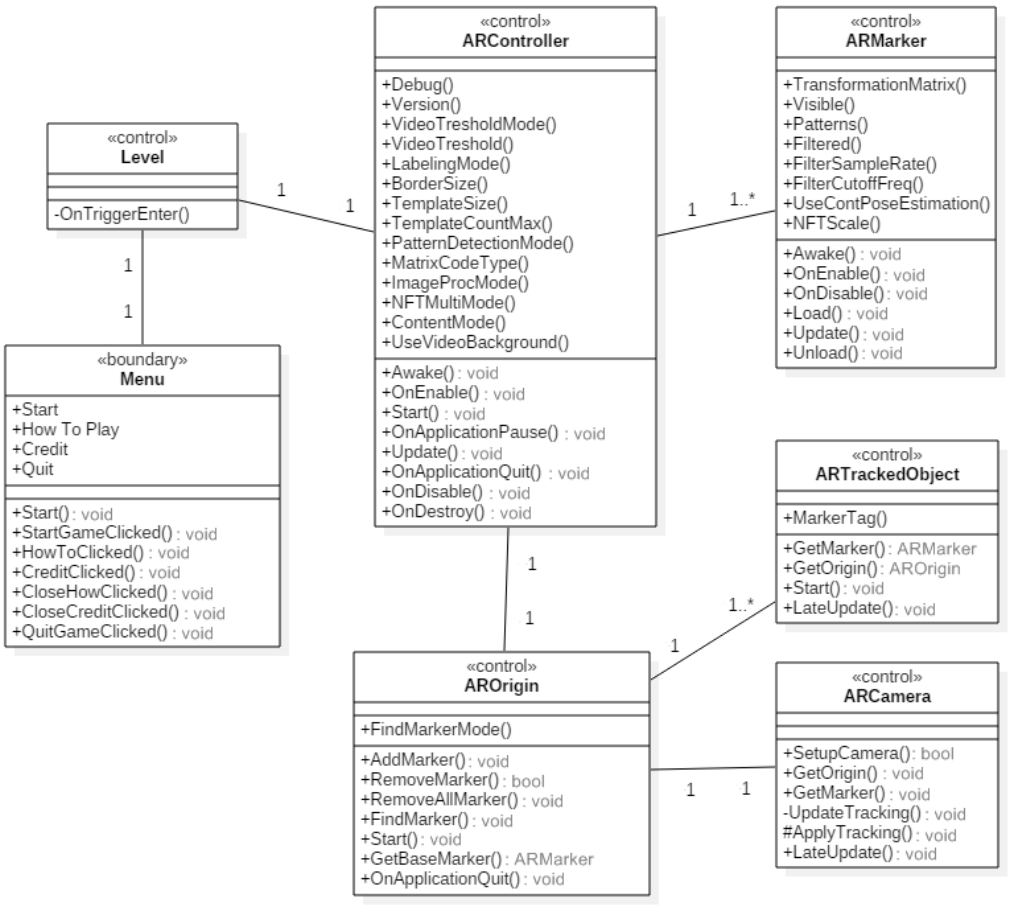
Activity diagram memainkan *game* menjelaskan tentang aktivitas cara memainkan *game* setelah *marker* terdeteksi oleh sistem. *Activity diagram* memainkan *game* dapat dilihat pada gambar III.22.



Gambar III. 22 *Activity Diagram* Memainkan *Game*

III.4.4 *Class Diagram*

Class diagram merupakan sebuah spesifikasi yang jika diinisialisasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan *desain* berorientasi objek. *Class* merupakan menggambarkan keadaan suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut. *Class diagram* menggambarkan struktur dan deskripsi *class*, *package* dan objek beserta hubungan satu sama lain. Berikut adalah *class diagram* pada aplikasi:



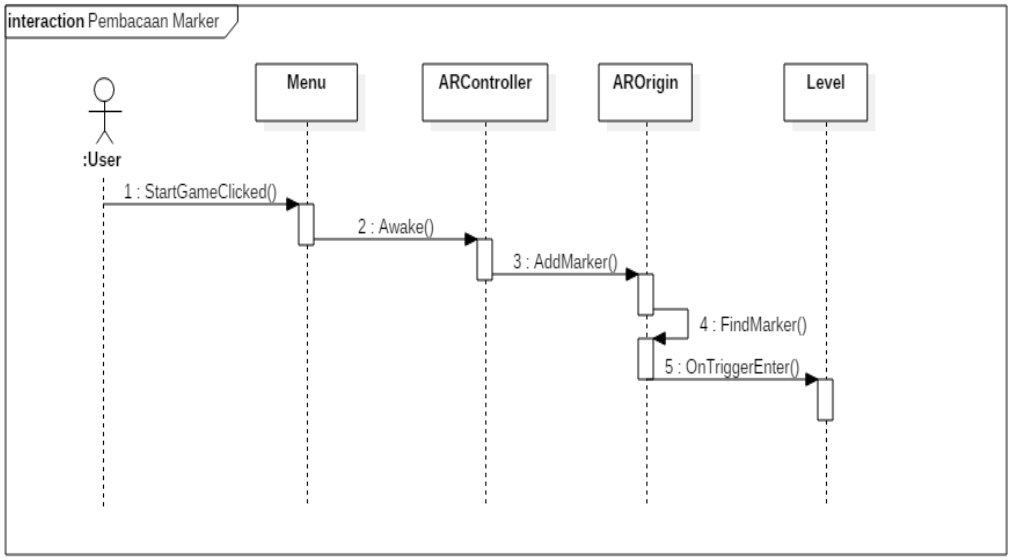
Gambar III. 23 *Class Diagram*

III.4.5 *Sequence diagram*

Sequence diagram menggambarkan interaksi antar masing-masing objek pada setiap *use case* dalam urutan waktu. Interaksi ini berupa pengiriman serangkaian data antar objek-objek yang saling berinteraksi. *Sequence diagram* pada sistem yang dibangun berikut ini dari masing-masing *Sequence diagram*:

1. *Sequence diagram* pembacaan *marker*

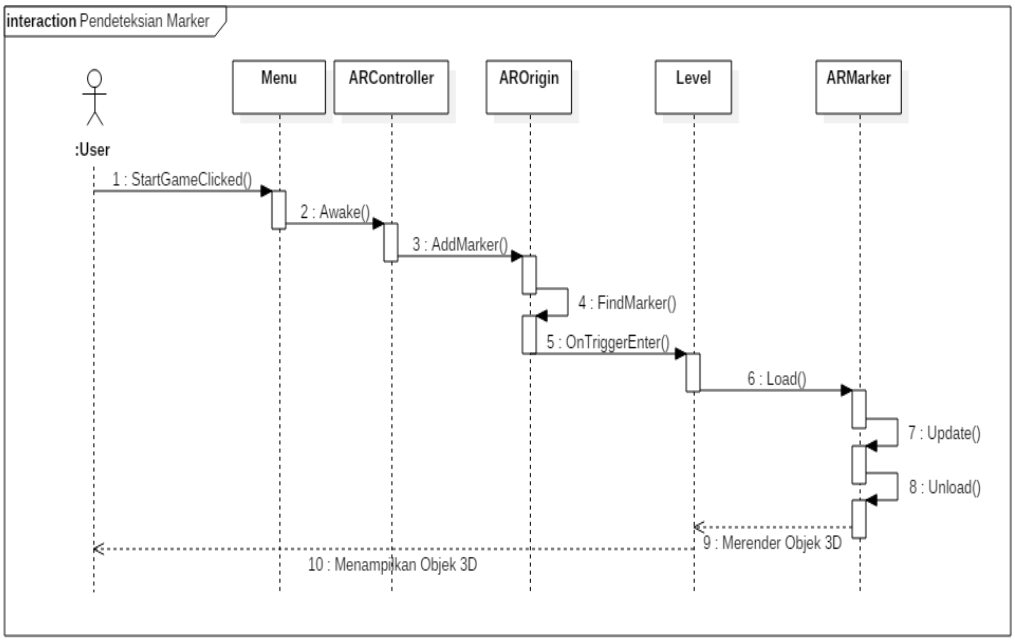
   Sequence diagram pembacaan *marker* dapat dilihat pada gambar III.24.



Gambar III. 24 *Sequence Diagram* Pembacaan *Marker*

2. *Sequence diagram* pendeteksian *marker*

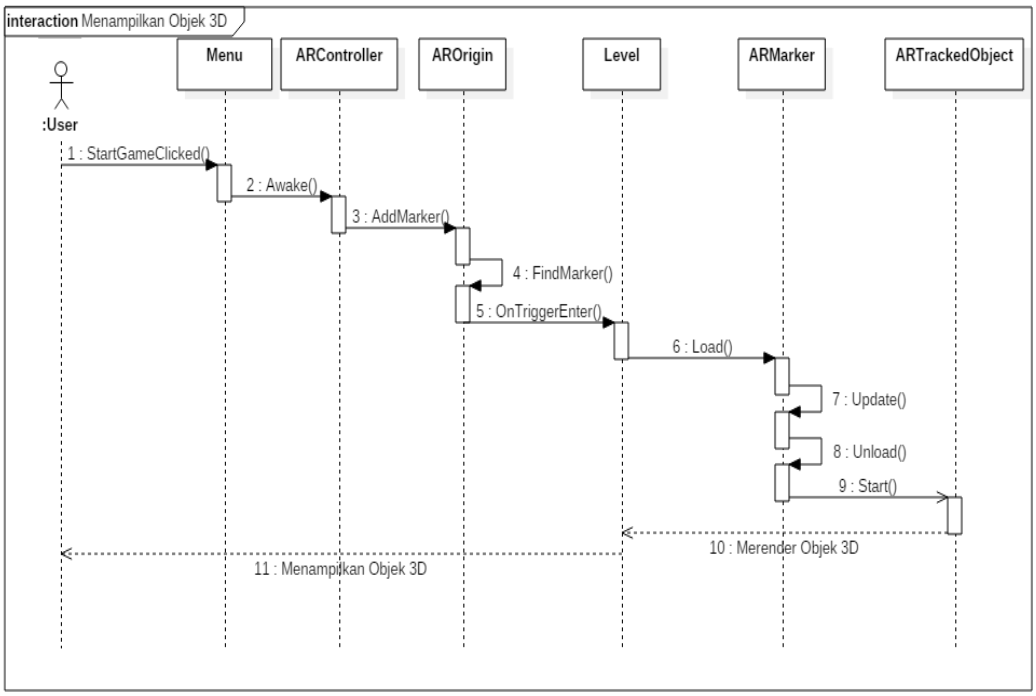
   Sequence diagram pendeteksian *marker* dapat dilihat pada gambar III.25.



Gambar III. 25 *Sequence Diagram* Pendeteksian *Marker*

3. *Sequence diagram* menampilkan objek 3d

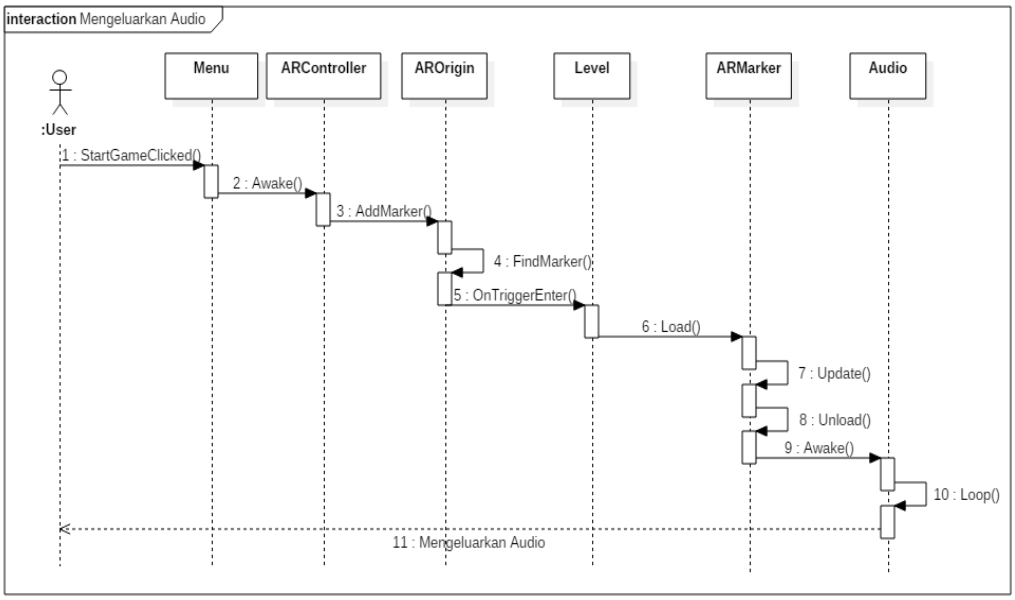
   Sequence diagram menampilkan objek 3d dapat dilihat pada gambar III.26.



Gambar III. 26 *Sequence Diagram* Menampilkan Objek 3D

4. *Sequence diagram* mengeluarkan *audio*

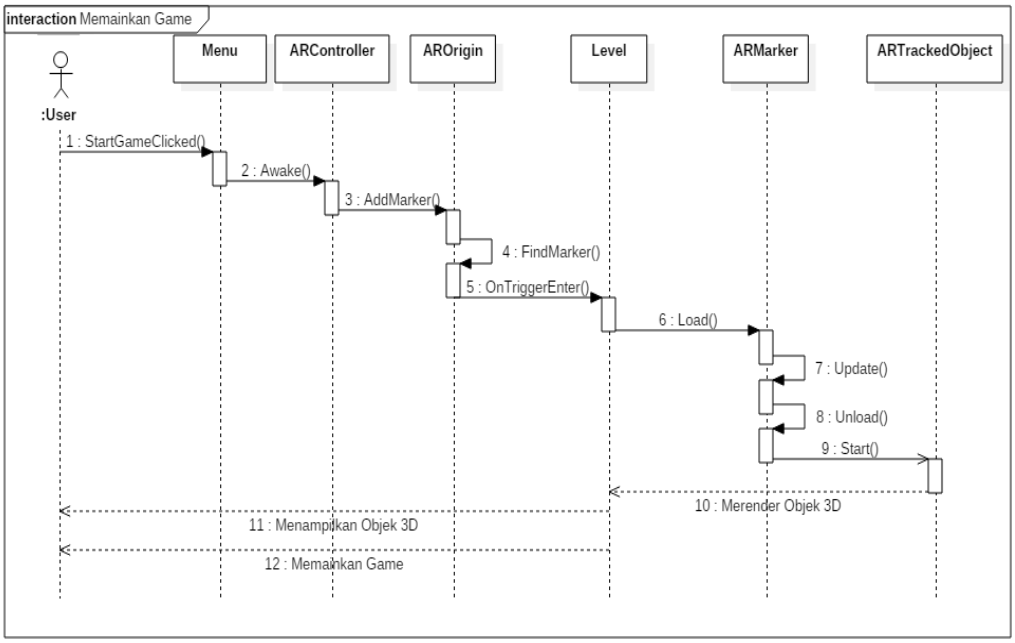
   Sequence diagram mengeluarkan *audio* dapat dilihat pada gambar III.27.



Gambar III. 27 *Sequence Diagram* Mengeluarkan *Audio*

5. *Sequence diagram* memainkan *game*

Sequence diagram memainkan *game* dapat dilihat pada gambar III.28.



Gambar III. 28 *Sequence Diagram* Memainkan *Game*

III.5 Analisis Kebutuhan Non Fungsional

Analisis kebutuhan non fungsional berisi tentang kebutuhan-kebutuhan pendukung yang diperlukan saat pembangunan aplikasi. Berikut adalah kebutuhan *non* fungsional untuk membangun aplikasi:

III.5.1 Analisis Kebutuhan Perangkat Keras

Pada aplikasi *game augmented reality* yang akan dibangun dibutuhkan perangkat keras untuk membantu pembangunan aplikasi. Adapun perangkat keras yang dibutuhkan adalah sebagai berikut:

a. *Processor* : Intel Celeron CPU N2840 2.16 Ghz
b. *Memory* : RAM DDR3 1 GB
c. VGA : 512 GB
d. *Harddisk* : 120 GB
e. Monitor : 14 *inch*
f. Mouse
g. Keyboard

III.5.2 Analisis Kebutuhan Perangkat Lunak

Perangkat lunak merupakan salah satu elemen penting dalam pembangunan *game* ini, karena perangkat lunak merupakan elemen yang digunakan dari mulai pembangunan *game* sampai dengan *game* selesai dibangun. Perangkat lunak yang dibutuhkan dalam pembangunan *game* ini adalah:

a. Sistem operasi yang digunakan dalam pembangunan *game* ini adalah windows 7.
b. *Software* yang digunakan adalah Unity 5.2.0f3, untuk pembangunan *game* nya.

III.5.3 Analisis Pengguna

Aplikasi yang akan dibangun tentunya memerlukan pengguna untuk menjalankannya, adapun spesifikasi pengguna yang dibutuhkan untuk menjalankan aplikasi tersebut adalah bisa mengoperasikan komputer. Untuk karakteristik kebutuhan pengguna dalam aplikasi ini dapat dilihat dalam tabel III.8 berikut.

Tabel III. 8 Analisis Kebutuhan Pengguna

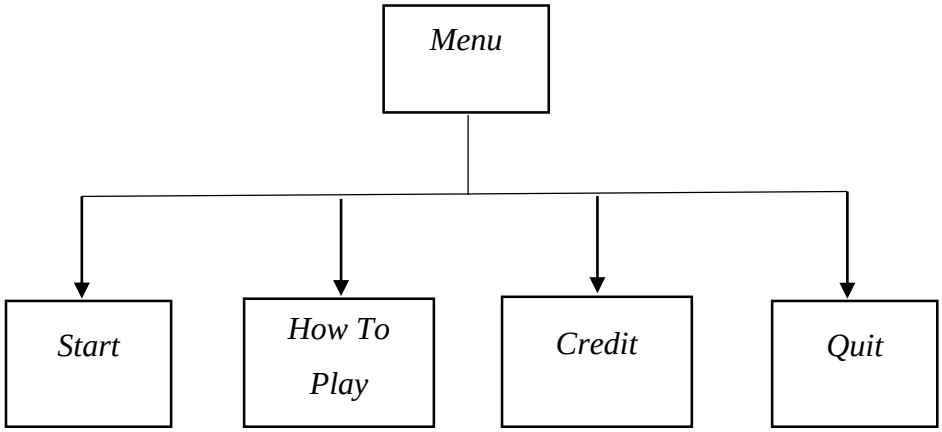
Pengguna	**Hak Akses**	**Tingkat Pendidikan**	**Pengalaman**
Semua usia	Dapat menjalankan semua fitur yang terdapat didalam *game*	Tidak dibatasi	Dapat mengoprasikan komputer

III.6 Perancangan Sistem

Tahap perancangan sistem merupakan tahap lanjutan dari tahap analisis terhadap sistem selesai. Tahap perancangannya dapat didefinisikan sebagai tahap perencanaan dan pembuatan untuk mendapatkan gambaran dari aplikasi yang akan dibangun. Adapun tahap perancangan sistem ini meliputi perancangan struktur *menu*, perancangan antarmuka dan jaringan semantik.

III.6.1 Perancangan Struktur *Menu*

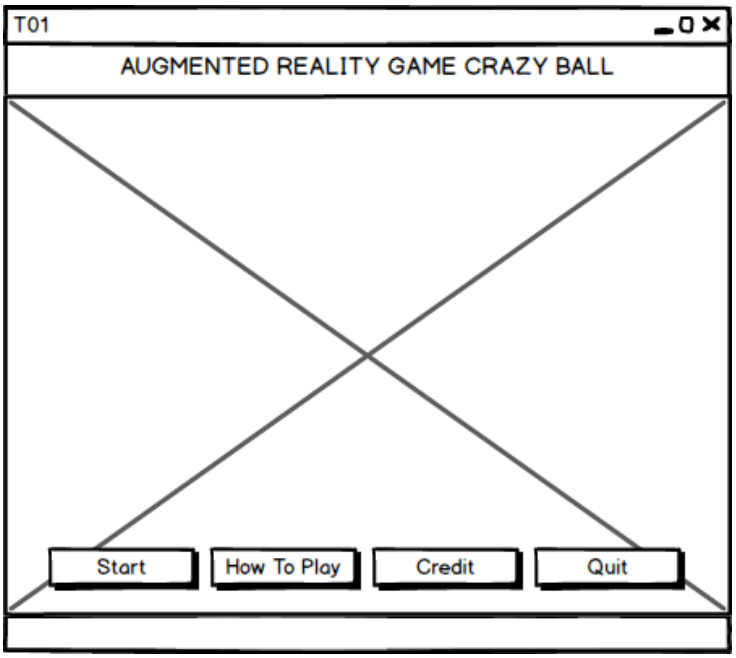
Perancangan struktur *menu* dan *submenu* yang akan berfungsi untuk memudahkan pengguna dalam menggunakan sistem. Struktur *menu* untuk sistem yang akan dibangun dapat dilihat pada gambar III.30.



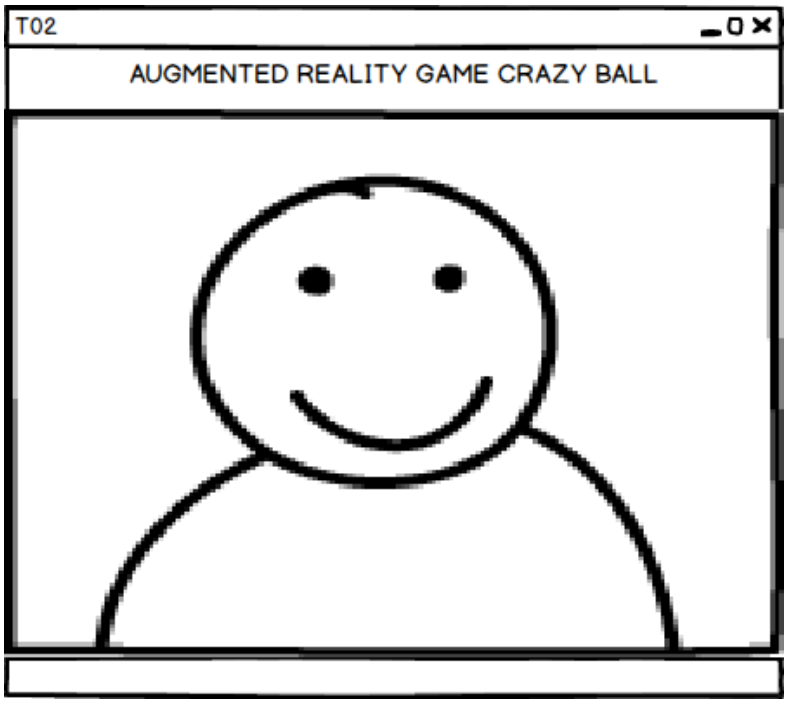
Gambar III. 29 Struktur *menu* sistem

III.6.2 Perancangan Antarmuka

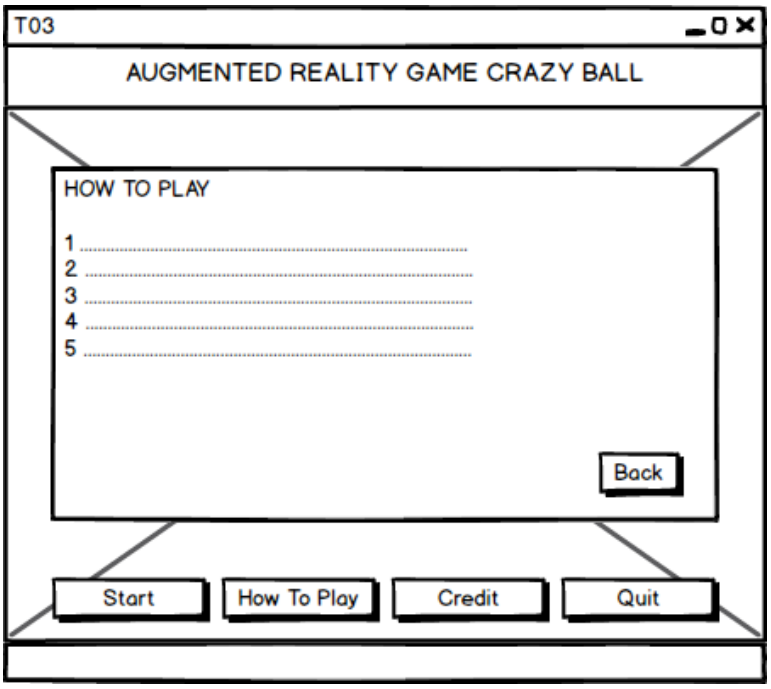
Perancangan antarmuka merupakan tahap perancangan untuk menggambarkan tampilan dari sistem yang akan dibangun. Perancangan antarmuka pada aplikasi *augmented reality* ini adalah sebagai berikut :



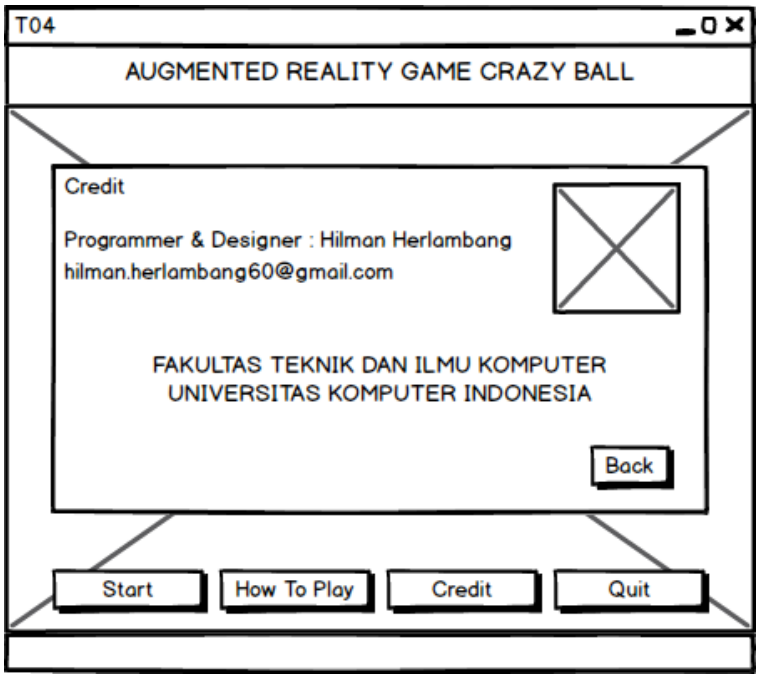
Gambar III. 30 Tampilan *Menu*



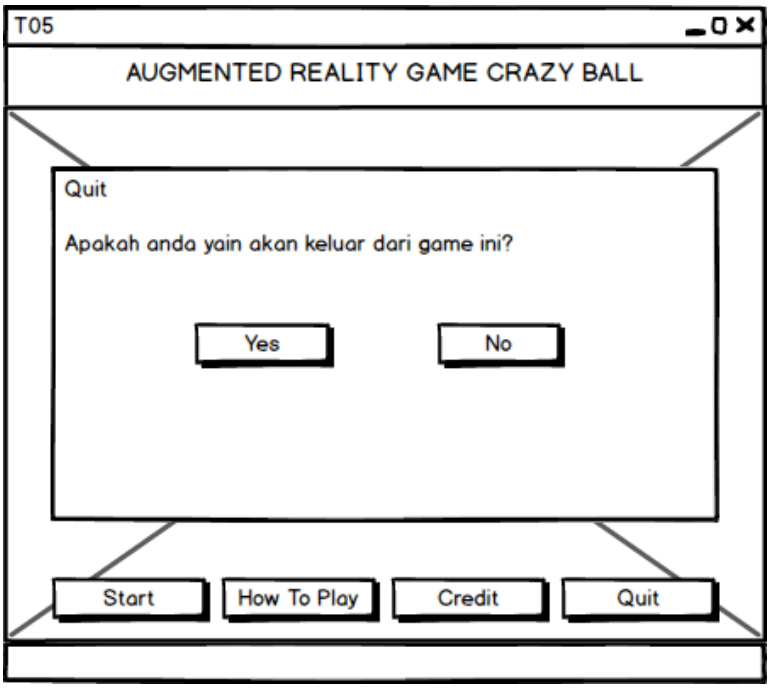
Gambar III. 31 Tampilan *Start*



Gambar III. 32 Tampilan *How To Play*



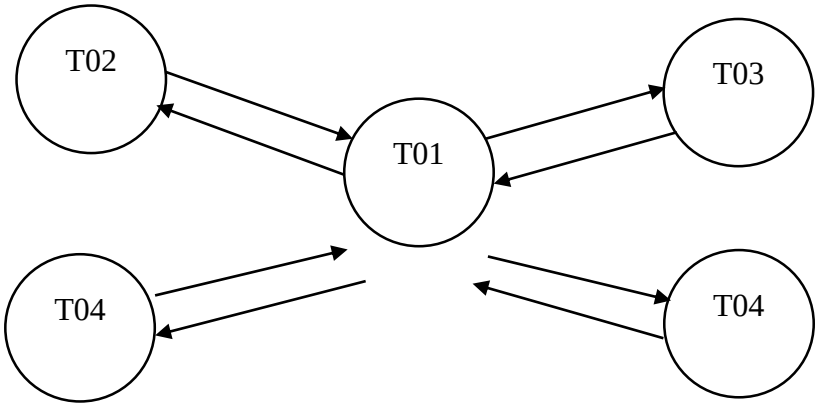
Gambar III. 33 Tampilan *Credit*



Gambar III. 34 Tampilan *Quit*

III.6.3 Jaringan Semantik

Jaringan semantik adalah gambaran yang menunjukkan hubungan antar objek yang terdapat pada sistem. Berikut adalah jaringan semantik yang menjelaskan hubungan dari setiap *menu* yang terdapat pada *game augmented reality*:



Gambar III. 35 Jaringan Semantik