BAB 3
ANALISIS DAN PERANCANGAN SISTEM

3.1 Analisis Masalah

Analisis masalah yang dimaksud adalah berkurangnya ketertarikan dan minat anak dalam kegiatan belajar mewarnai saat ini dikarenakan kurangnya dampak nyata yang didapat oleh anak sehingga membuat anak menjadi tertarik terhadap kegiatan belajar mewarnai. Seiring dengan perkembangan teknologi, anak cenderung lebih tertarik terhadap sesuatu yang lebih interaktif dan memberikan dampak langsung terhadap kegiatan mereka. Saat ini proses kegiatan belajar mewarnai hanya dilakukan dengan memanfaatkan buku mewarnai dan alat tulis warna. Dibutuhkan sebuah aplikasi yang dapat menjadi media tambahan bagi anak dalam melakukan kegiatan belajar mewarnai dengan memanfaatkan teknologi yang diminati oleh anak dan dapat digunakan dimana saja dan kapan saja. Dengan demikian kegiatan belajar mewarnai yang dilakukan dapat menjadi lebih menarik dan dapat dilakukan tidak hanya saat berada di sekolah.

3.1.1 Analisis Kondisi Saat Ini

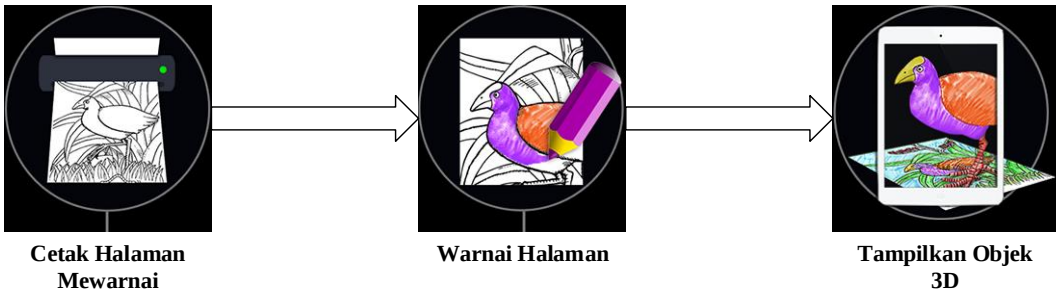
Analisis kondisi saat ini bertujuan untuk memberikan gambaran proses kegiatan belajar mewarnai yang ada. Saat ini, proses kegiatan belajar mewarnai dilakukan dengan media buku mewarnai yang didalamnya terdapat bermacam-macam objek mulai dari objek dasar sampai objek yang ada disekitar seperti bola, tanaman, dan hewan. Anak-anak bertugas untuk mewarnai objek-objek tersebut sesuai dengan kreativitas mereka masing-masing. Tugas dari orang tua/guru adalah untuk mengawasi dan memberikan pengarahan dari kegiatan mewarnai tersebut.

3.1.2 Usulan Aplikasi yang Akan Dibangun

Usulan aplikasi yang akan dibangun adalah suatu aplikasi *mobile* menggunakan platform Android dengan teknologi *Augmented Reality*. Teknologi dengan aplikasi *mobile* dengan mudah diakses dimana saja dan sangat berkembang pada masa kini sehingga memudahkan pengguna untuk mengakses aplikasi ini.

Keunggulan dari teknologi *augmented reality* juga sangat menonjol dalam segi menampilkan suatu informasi secara *realtime* sehingga dapat diterapkan kedalam aplikasi, untuk dapat memberikan solusi dari permasalahan yang ada.

Aplikasi ini merupakan aplikasi yang mengkolaborasikan antara proses kegiatan mewarnai biasa dengan teknologi *augmented reality*. Proses *augmented reality* dilakukan dengan melakukan proses mewarnai pada bidang mewarnai seperti biasa kemudian bidang mewarnai yang telah dijadikan marker tersebut diarahkan ke kamera pada perangkat Mobile untuk menampilkan objek 3D yang memiliki tekstur sama seperti pada bidang marker yang diwarnai. Rancangan aplikasi *augmented reality* yang akan dibangun dapat dilihat seperti pada Gambar 3.1.



Gambar 3.1 Usulan Aplikasi yang Dibangun

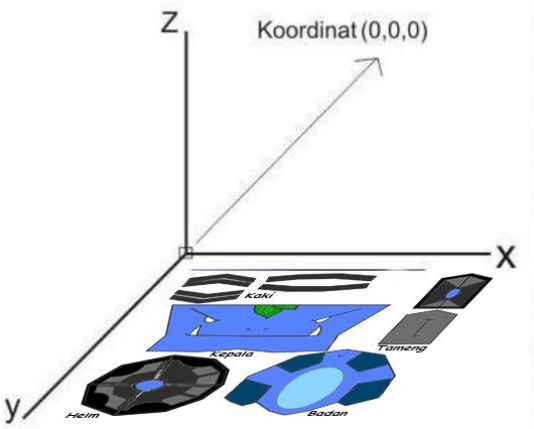
3.2 Analisis Kebutuhan Data

Aplikasi yang dibangun adalah aplikasi *Magic Coloring Book* dengan memanfaatkan teknologi *augmented reality*. Data masukan yang dibutuhkan antara lain berupa marker dan objek 3D sebagai input utama pada aplikasi. Pada proses *augmented reality*, aplikasi akan memunculkan objek tiga dimensi jika aplikasi mendeteksi marker yang diarahkan pada kamera, sehingga aplikasi ini membutuhkan *input* berupa citra yang diambil secara *real-time* menggunakan alat *input* berupa kamera dimana citra yang diambil tersebut merupakan kumpulan matriks yang nantinya akan diolah lebih lanjut dalam sistem untuk proses pendeteksian marker untuk memunculkan objek tiga dimensi yang memiliki tekstur yang sama dengan tekstur pada marker yang dideteksi.

3.2.1 Analisis Marker

Marker merupakan perangkat keras lainnya yang merupakan salah satu bagian dari *Augmented Reality* (AR) yang digunakan untuk membuat suatu aplikasi *Augmented Reality* (AR), pada umumnya marker merupakan sebuah persegi hitam dan di tengahnya terdapat bentuk atau ciri yang membedakan antara satu marker dengan marker lain yang akan menjadi *keypoint*, marker yang terdapat ditengah bisa berbentuk gambar, angka, huruf, atau apa saja, didalam pola marker yang merupakan ilustrasi hitam dan putih dengan batas hitam yang tebal dan latar belakang putih digunakan agar komputer mengenali posisi dan orientasi marker dan menciptakan dunia virtual 3D yaitu pada titik (0,0,0) dari 3 sumbu yaitu X,Y, dan Z.

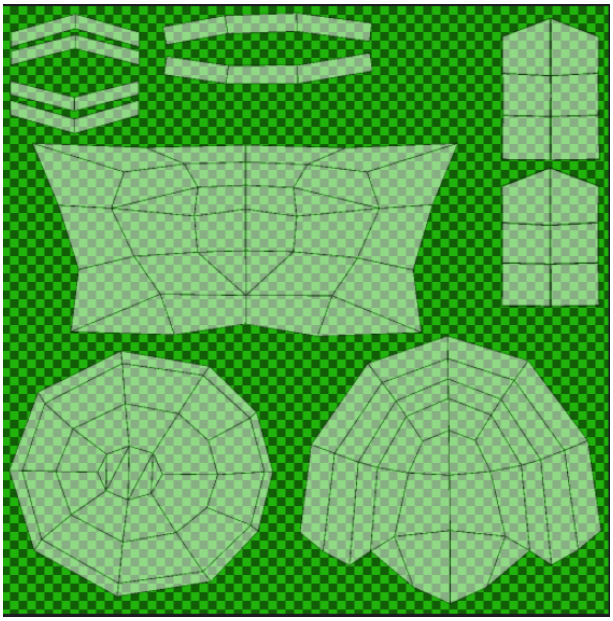
Pada aplikasi ini, objek marker yang digunakan berupa objek gambar yang akan dijadikan media mewarnai. Objek-objek tersebut akan merepresentasikan objek 3D yang akan ditampilkan dalam proses *augmented reality*. Warna yang ada pada objek marker ini pula yang akan menjadi tekstur warna pada objek 3D yang ditampilkan. Objek marker sekaligus media mewarnai yang dibuat berupa pecahan-pecahan bagian dari sebuah objek. Hal ini bertujuan untuk mempermudah pembacaan tekstur sekaligus meningkatkan daya imajinasi dari anak untuk mengkreasikan pecahan objek yang nantinya akan direpresentasikan dalam bentuk *augmented* 3D. Contoh dari objek marker sekaligus objek mewarnai yang dibuat dapat dilihat seperti pada Gambar 3.2.



Gambar 3.2 Marker beserta koordinatnya

3.2.1.1 Analisis Gambar Marker

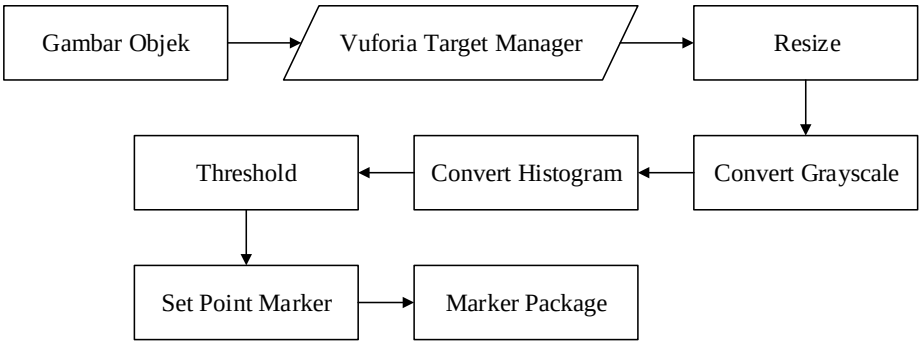
Gambar marker yang dibuat merupakan hasil dari UVW Mapping dan *texture mapping* terhadap objek 3D. Hal ini bertujuan agar saat proses pengambilan tekstur nantinya, tekstur dari gambar marker yang ditangkap dapat langsung diubah menjadi format tekstur 2D dan diaplikasikan menjadi tekstur dari objek 3D yang ditampilkan pada proses *augmented reality* aplikasi. Proses ini dilakukan dengan cara mengekstrak permukaan dari objek 3D menggunakan aplikasi Blender. Contoh hasil UVW Mapping dan *texture mapping* dari objek 3D dapat dilihat pada Gambar 3.3.



Gambar 3.3 UVW Mapping dan *Texture Mapping* Objek 3D

3.2.1.2 Pembuatan Marker

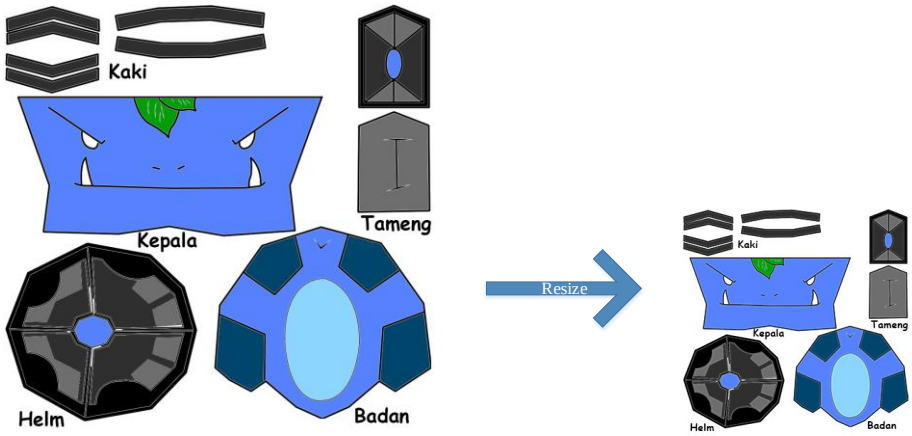
Proses pembuatan gambar marker menjadi marker dilakukan oleh pihak Vuforia dengan cara mengunggah gambar yang akan diubah, melalui marker *engine* yang disediakan oleh pihak Vuforia. Setelah gambar di ubah menghasilkan *file* dengan format .unitypackage. *File* tersebut kemudian dijadikan masukan pada Engin Unity3D untuk mendeteksi gambar yang dijadikan marker. Berikut tahapan atau alur dalam pembuatan marker seperti terlihat pada Gambar 3.4.



Gambar 3.4 Alur Proses Pembuatan Marker

1. Resize

Setelah kita mengunggah gambar ke target manager di situ vuforia developer, maka gambar yang telah kita unggah akan melalui proses resize atau penskalaan gambar secara otomatis dengan lebar menjadi 320px dan tinggi menyesuaikan dengan rasio gambar. Gambar 3.5 menunjukkan citra ukuran citra asli dan ukuran citra setelah proses resize.



Gambar 3.5 Resize Gambar

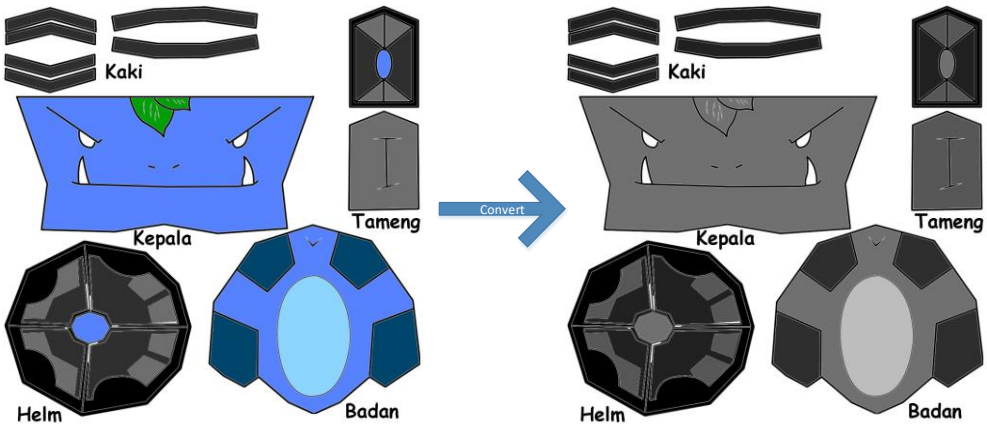
Algoritma resize yang digunakan adalah seperti berikut:

```
Read(image,imgx,imgy)
Max_x<- 250 (maksimal nilai lebar gambar) Max_y<250(maksimal
nilai panjang gambar)
If imgx>=Max_x
then Scale = Max_x div imgx
new_imgx = imgx * scale
new_imgy = imgy *scale endif
```

```
If new_imgy > Max_y then
Scale = Max_y/new_imgy
new_imgx = new_imgx*scale
new_imgy = new_imgy*scale
endif
If imgx<Max_x Then new_imgx = imgx;
If imgy<Max_y Then new_imgy = imgy;
```

2. Convert Grayscale

Grayscale merupakan proses mengubah gambar menjadi monokrom dengan nilai intensitas 0 untuk hitam, 255 untuk putih, dan abu-abu untuk nilai yang berada pada rentang nilai 0 sampai 255. Dimana rumus secara garis besarnya adalah new pixel = (Red + Green +Blue)/3. Gambar 3.6 menunjukkan gambar citra asli dan gambar citra setelah dilakukan proses grayscale.



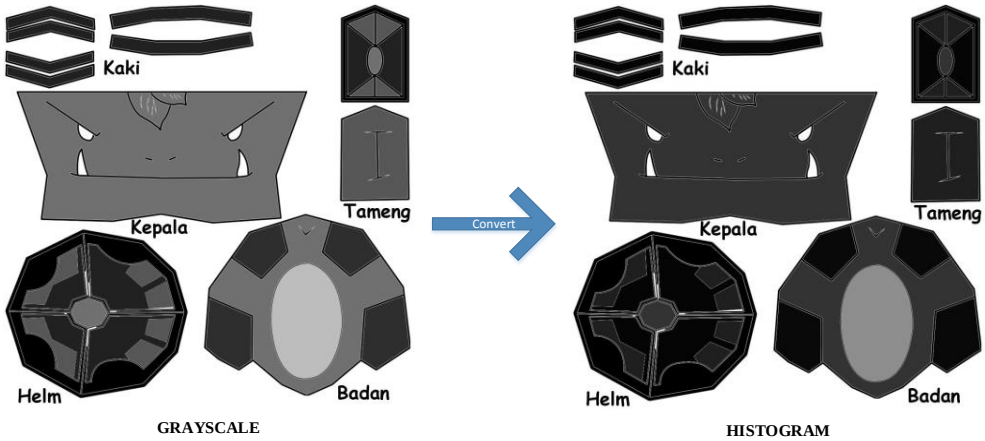
Gambar 3.6 Convert Grayscale

Berikut adalah algoritma yang digunakan dalam proses grayscale:

```
Read (image,x,y)
For (i<-0 ; i<x; i++) do
   For (j<-0; j<y; j++) do
         Getpixel(i,j);
         Int grayscale = (int)((originalcolor.R)+
         (originalcollor.G)+(originalcolor.B)/3)
         Newimage(i,j,grayscale)
   End for
End for
```

3. Convert Histogram

Proses Histogram adalah suatu proses perataan, dimana distribusi derajat keabuan pada suatu gambar dibuat rata dapat dilihat seperti Gambar 3.7.



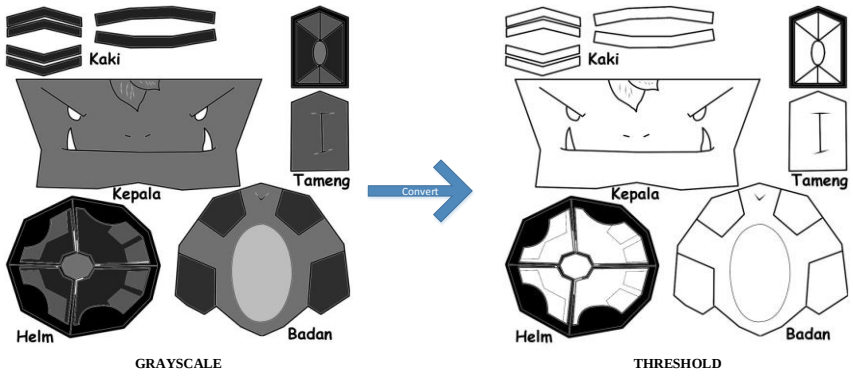
Gambar 3.7 Proses Histogram

Algoritma untuk proses histogram adalah sebagai berikut:

```
Deklarasi : (A,int N,int M)
/*A=array citra N, M=panjang,Lebar gambar*/
/*inisialisasi Hist[0,255] dengan 0*/
for(i=0;i<=255;i++)
   Hist[i]=0
Endfor
/*mencacah frekuensi kemunculan*/
for(k=0;k<=255;k++)
   for (i=0;i<=N-1;i++)
         for(j=0,j<=M-1;j++)
             if(A[i,k]==i-1)
                Hist[i]=Hist[i]+1
             Endif
         Endfor
   Endfor
Endfor
```

4. Threshold

Dalam Library Vuforia sebelum objek diconvert kedalam point-point marker yang dapat menghasilkan titik sudut marker, marker diubah kedalam bentuk Threshold. Proses threshold ini yang menjadi acuan untuk proses selanjutnya, yaitu proses penentu titik marker. Berikut contoh perubahan gambar dari gambar asli ke gambar Threshold Set Marker Point seperti terlihat pada Gambar 3.8.



Gambar 3.8 Proses Threshold

Berikut algoritma yang digunakan dalam proses thresholding:

```
Tmean = 0 /*deklarasi nilai awal t*/
For (k = 0; k < level; k++)
For x=0 to panjang_pixel-1 do
For y=0 to lebar_pixel-1 do
N= image[i],[y]
If n<=T then
new_x=0
new_y=0
t=k;
else
new_x=255
new_y=255
endif
endfor
endfor
endfor
```

3.2.2 Analisis Objek 3D

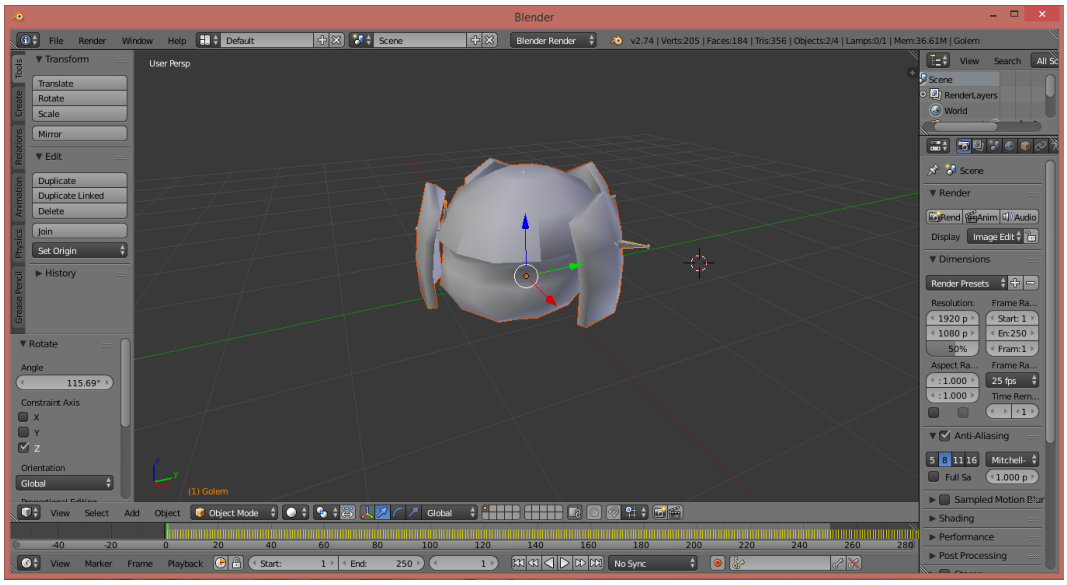
Model 3D yang akan ditampilkan di-*load* terlebih dahulu pada *engine* Unity3D. Agar aplikasi dapat menampilkan objek 3D tertentu tanpa merubah atau membangun ulang aplikasi, diperlukan sebuah *file* format .OBJ atau .FBX untuk menentukan objek 3D yang akan di-*load* di Unity3D. Pembuatan objek 3D sampai dengan riggingnya menggunakan *software* Blender, setelah di buat lalu diekspor ke format .OBJ atau .FBX agar dapat di-*eksport* melalui Unity3D. Berikut tahapan inisialisasi model 3D seperti pada Gambar 3.9.



Gambar 3.9 Proses Inisialisasi Model 3D

3.2.2.1 Pembuatan Objek 3D

Objek 3D merupakan objek yang dijadikan sebuah *output* yang hasilkan dihasilkan pada tahap proses *tracking* marker. Dalam analisis objek 3D yang digunakan pada *vuforia* harus berupa objek 3D yang berekstensi file .fbx dan .obj sehingga *file* 3D tersebut dapat ditampilkan pada unity engine Gambar 3.10 menunjukkan pembuatan objek 3D dengan menggunakan *software* Blender.



Gambar 3.10 Pembuatan Objek 3D dengan Blender

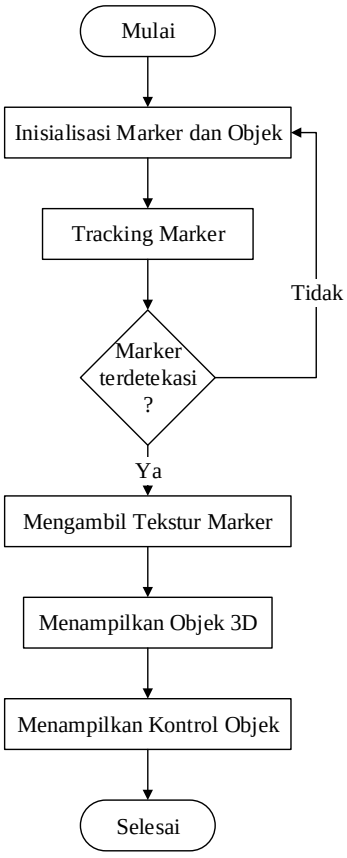
Dalam proses pembentukan objek 3D ada beberapa langkah yang dilakukan :

1. Menyesuaikan objek 3D yang dibuat dengan menambahkan tekstur yang dibutuhkan.
2. Mengekspor objek 3D yang telah dirancang menjadi format .fbx Pada tahap mengekspor objek 3D dapat dilakukan dengan menggunakan *software* blender itu sendiri.

3.3 Analisis Augmented Reality

Perancangan aplikasi *Augmented Reality*, menggabungkan objek virtual atau objek 3D dengan objek nyata atau (*marker*). Secara keseluruhan, proses dari *Augmented Reality* dimulai dari webcam atau kamera yang sudah dikalibrasi akan mendeteksi *marker* yang diberikan. Kemudian setelah mengenali dan menandai

pola *marker*, perangkat kamera akan melakukan pengecekan apakah *marker* yang terdeteksi sesuai dengan data *marker* yang dimiliki oleh aplikasi. Bila tidak, maka objek 3D tidak akan ditampilkan, tetapi bila sesuai maka objek 3D akan ditampilkan di atas *marker*. Dalam penelitian ini proses-proses aplikasi *Augmented Reality* yang dibangun dibagi menjadi tahapan-tahapan yang dapat dilihat seperti pada Gambar 3.11.



Gambar 3.11 Tahapan Aplikasi Augmented Reality

Secara garis besar, tahapan-tahapan yang terdapat pada aplikasi *augmented reality* ini dibagi menjadi 3 bagian, yaitu:

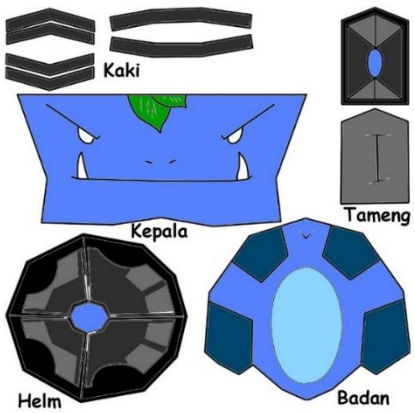
1. Tahap inisialisasi objek dan marker
2. Tahap Tracking Markerless
3. Tahap Rendering Objek

3.3.1 Inisialisasi Marker dan Objek

Tahap ini aplikasi *Augmented Reality* akan melakukan inisialisasi *windows* atau layar berupa warna transparan. Setelah itu aplikasi akan dihubungkan dengan tampilan kamera untuk memulai proses pendeteksian *marker*. Kemudian aplikasi akan melakukan inisialisasi variabel awal yang dipakai untuk menampung sementara *marker*, status pengecekan, serta variabel informasi yang akan ditampilkan. Spesifikasi pola penanda atau *marker*:

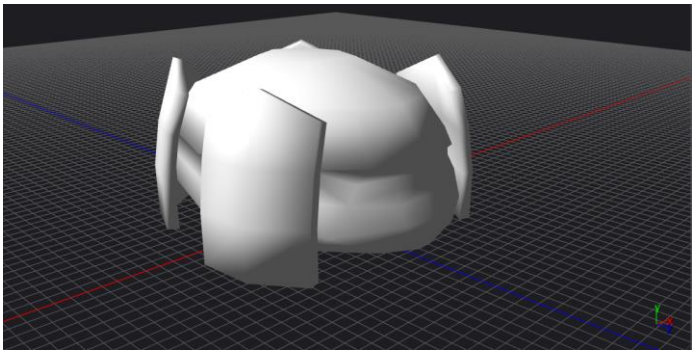
1. Pola penanda minimum memiliki lebar 320 piksel.
2. Format dari gambar yang diunggah ke situs vuforia berformat .jpg atau .png

Contoh *marker* yang digunakan dapat dilihat pada Gambar 3.12.



Gambar 3.12 Contoh Marker

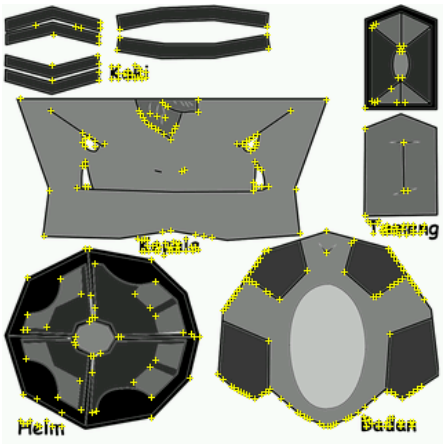
Marker yang didaftarkan sengaja dibuat dalam bentuk yang berwarna dikarenakan akan mempermudah pengenalan pola sehingga memiliki nilai *rate* bintang bernilai 5. Sehingga sudah cukup untuk dilacak oleh sistem AR. Selain menginisialisasi *marker*, model 3D juga dimuat dahulu pada *engine* Unity untuk dipasangkan dengan *marker* yang bersesuaian. Objek 3D yang digunakan berformat .3DS dan .FBX agar dapat dieksport ke Unity. Gambar dari model 3D yang digunakan pada aplikasi dapat dilihat pada Gambar 3.13.



Gambar 3.13 Contoh Model 3D yang Digunakan

3.3.2 Tracking Marker

NFT merupakan metode yang digunakan di dalam Library Vuforia Qualcomm untuk mendeteksi pola gambar. Sebelum dilakukannya pendeteksian fitur-fitur pada gambar, terlebih dahulu dilakukan proses identifikasi *marker*. Identifikasi *marker* adalah proses untuk menangkap gambar yang ditangkap oleh kamera. Gambar yang diperoleh itu dianggap *marker* terlepas ada atau tidaknya *marker* yang sesungguhnya pada gambar. Setelah *marker* diperoleh maka proses selanjutnya adalah mengubah *marker* menjadi keabu-abuan atau proses *grayscale*. Proses ini dilakukan untuk memudahkan dalam pendeteksian fitur pada *marker*. *Marker* yang sudah diubah menjadi keabu-abuan kemudian dideteksi fiturnya menggunakan metode NFT. Metode yang digunakan di dalam NFT itu sendiri adalah SIFT dan FERNS. Hasil *output* gambar dari perhitungan dapat dilihat pada Gambar 3.14.



Gambar 3.14 Gambar Hasil Deteksi Fitur Lokal

Komponen-komponen dalam proses deteksi fitur yang dilakukan adalah sebagai berikut :

1. *Trackable Type*
   a. UNKNOWN_TYPE : Pelacakan yang tidak diketahui
   b. IMAGE_TARGET : Pelacakan berdasarkan gambar
   c. MULTI_TARGET : Pelacakan lebih dari 1 target
   d. MARKER : Pelacakan *marker*
2. *Trackable Name*

   Pelacakan menggunakan kumpulan karakter unik yang disusun sebagai target identifikasi pelacakan. Untuk penulisan nama hanya diperbolehkan maksimal 64 karakter dan hanya mengandung karakter (a-z, A-Z, 0-9, [-_.]).
3. *Trackable Status*

   Informasi status pelacakan yang dilakukan per-*frame* dan akan selalu di-*update* secara *real-time*. Status-status yang dapat terjadi pada saat pelacakan adalah sebagai berikut :
   a. UNKNOW : gambar *marker* pada pelacakan tidak diketahui biasanya dikembalikan sebelum *tracker initialization*
   b. UNDEFINE : gambar *marker* tidak didefinisikan
   c. NOT_FOUND: gambar *marker* tidak ditemukan pada *library* yang dituju
   d. DETECTED : gambar *marker* terdeteksi dalam *frame*
   e. TRACKED : pelacakan telah terlacak dalam *France*

Beberapa fungsi yang digunakan pada vuforia dalam proses tracking marker adalah sebagai berikut:

CameraDevice

Fungsi kelas ini untuk menyediakan akses ke kamera dan properti.

QCARBehaviour

Merupakan kelas untuk menangani pelacakan dan memicu tampilan *background rendering* video pada kamera.

DataSetLoadBehaviour

Fungsi ini memungkinkan untuk secara otomatis memuat dan mengaktifkan satu atau lebih *data set* ketika *startup* pada *image target*.

```
TrackbleBehaviour.Trackable [get]
```

Pelacakan pada saat dijalankan.

```
TrackbleBehaviour.Trackable [get]
```

Berfungsi sebagai register baru Tracker Event Handler pada *tracker*. Penanganan ini dilakukan setelah semua Trackable telah diperbarui.

```
Status TrackableBehaviour.CurrentStatus [get]
```

Status pelacakan dari TrackableBehaviour.

```
TrackableBehaviour.UnregisterTrackableEventHandler
```

Berfungsi sebagai *unregisters* sebuah Tracker Event Handler mengembalikan nilai "false" jika *event handler* tidak ada.

```
String TrackableBehaviour.TrackableName [get]
```

Pelacakan nama yang ada pada TrackableBehaviour.

```
void TrackableBehaviour.OnTrackerUpdate (Status newStatus)
[inline, virtual]
```

Dipicu oleh TrackerBehaviour setelah itu diperbarui.

```
DefaultTrackableEventHandler.OnTrackableStateChanged
```

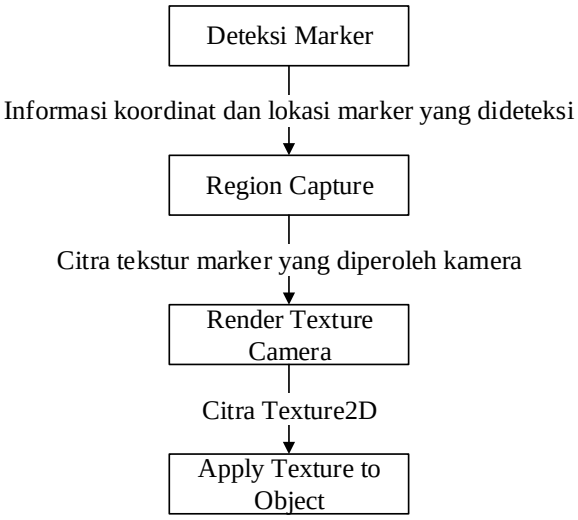
Sebuah *handler custom* yang mengimplementasikan antarmuka ITrackableEventHandler.

```
DefaultInitializationErrorHandler
```

Fungsi untuk menginisialisasi eror pada DevaultTrackableEventHandler

3.3.3 Pengambilan Tekstur Marker

Tahap ini merupakan tahap dimana tekstur dari marker yang telah diwarnai akan dideteksi untuk mengetahui batas wilayah/*region* dari marker tersebut untuk kemudian di-*capture* agar diperoleh citra tekstur marker. Citra tekstur yang diambil ini kemudian diproses/direnderuntuk mendapatkan citra tekstur yang dapat diaplikasikan pada objek 3D yang akan ditampilkan. Tahap pengambilan tekstur marker ini dapat dilihat seperti pada Gambar 3.15.



Gambar 3.15 Alur Tahap Pengambilan Tekstur Marker

Pada tahap awal, sistem akan membaca dan mendeteksi lokasi serta koordinat marker yang akan digunakan sebagai batas dari pengambilan citra tekstur marker. Setelah marker terdeteksi dan lokasi serta batas marker didapat, maka dilakukan proses pengambilan citra tekstur marker menggunakan fungsi `region_capture` seperti berikut:

```
if ((ARCamera or ImageTarget or BackgroundPlane) = null)
/*proses pendeteksian kamera dan target marker*/
then region_capture = false
else
transform.position = ImageTarget.transform.position;
transform.localScale = ImageTarget.transform.localScale / 10.0f;
/*penentuan dan penskalaan lokasi tekstur*/
GetComponent<Renderer>().material.SetFloat("_KX", k_x);
GetComponent<Renderer>().material.SetFloat("_KY", k_y);
/*pengambilan citra tekstur*/
```

Dari proses tersebut akan didapat citra tekstur marker yang kemudian akan diproses menjadi Texture2D yang merupakan GameObject yang dapat diaplikasikan sebagai tekstur pada objek 3D yang akan ditampilkan menggunakan proses `RenderTextureCamera` berikut:

```
Deklarasi : TextureResolution = 512; TextureResolutionX,
TextureResolutionY : int;
/*proses render tekstur dari kamera*/
RenderTexture CameraOutputTexture = new
RenderTexture(TextureResolutionX, TextureResolutionY, 0);
/*pembuatan objek tekstur baru dari tekstur yang diperoleh
kamera*/
CameraOutputTexture.Create();
Render_Texture_Camera.GetComponent<Camera>().targetTexture =
Render_Texture_Camera.cullingMask = 1 << 20;
/*proses pengaplikasian texture2D terhadap objek*/
DebugGUITexture.transform.localScale = Vector3(0.3f *
GuiTextureAspect, 0.3f, 0.3f);
```

3.4 Spesifikasi Kebutuhan Perangkat Lunak

Tahap ini menjelaskan analisis spesifikasi dari kebutuhan perangkat lunak yang akan dibangun yaitu meliputi spesifikasi kebutuhan perangkat lunak fungsional (SKPL-F) dan spesifikasi kebutuhan perangkat lunak non-fungsional (SKPL-NF).

1. Spesifikasi Kebutuhan Fungsional

   Spesifikasi kebutuhan funsional berisi mengenai fungsi atau layanan apa yang harus disediakan atau bisa dilakukan oleh aplikasi terhadap suatu kondisi tertentu, dan apa yang harus dilakukan oleh apliasi jika terdapat suatu *input* tertentu. Detail dari spesifikasi kebutuhan fungsional dapat dilihat pada Tabel 3.1.

Tabel 3.1 Spesifikasi Kebutuhan Perangkat Lunak Fungsional

Nomor	Spesifikasi Kebutuhan Perangkat Lunak Fungsional
SKPL-F-001	Aplikasi harus dapat membaca marker
SKPL-F-002	Aplikasi harus dapat mendeteksi marker
SKPL-F-003	Aplikasi harus dapat menampilkan objek 3D
SKPL-F-004	Aplikasi harus dapat mengambil tekstur marker
SKPL-F-005	Aplikasi harus dapat mengeluarkan audio
SKPL-F-006	Aplikasi harus dapat melakukan kontrol terhadap objek yang ditampilkan

2. Spesifikasi Kebutuhan Non Fungsional

   Spesifikasi kebutuhan non-fungsional merupakan penjelasan mengenai batasan-batasan fungsi dari aplikasi yang akan dibangun. Detail mengenai spesifikasi kebutuhan non-fungsional dapat dilihat pada Tabel 3.2.

Tabel 3.2 Spesifikasi Kebutuhan Perangkat Lunak Non Fungsional

Nomor	Spesifikasi Kebutuhan Perangkat Lunak Non Fungsional
SKPL-NF-001	Aplikasi memerlukan kamera dengan resolusi yang cukup untuk mengenali *marker*
SKPL-NF-002	Aplikasi memerlukan cahaya yang cukup agar kamera dapat mendeteksi *marker*
SKPL-NF-003	Aplikasi memelukan sudut yang cukup untuk membaca *marker* dari arah kamera

3.5 Analisis Kebutuhan Non Fungsional

Analisis kebutuhan non fungsional menggambarkan kebutuhan luar sistem yang diperlukan untuk menjalankan aplikasi yang dibangun. Adapun kebutuhan non fungsional pada aplikasi *Magic Coloring Book* dengan teknologi Augmented Reality ini meliputi kebutuhan perangkat keras, kebutuhan perangkat lunak dan pengguna sistem yang akan memakai aplikasi.

3.5.1 Analisis Kebutuhan Perangkat Keras

Analisis Perangkat keras atau *hardware* merupakan salah satu hal yang penting karena tanpa *hardware* yang memenuhi syarat, aplikasi yang akan dibuat tidak akan dapat berjalan. Analisis kebutuhan perangkat keras ini dibagi menjadi dua bagian, diantaranya seperti:

A. Analisis Kebutuhan Perangkat Keras Pembangunan Aplikasi

Kebutuhan perangkat keras yang digunakan untuk membangun aplikasi *Magic Coloring Book* dengan teknologi *Augmented Reality* dapat dilihat pada Tabel 3.3.

Tabel 3.3 Kebutuhan perangkat keras pembangunan aplikasi

No	Perangkat Keras
1	Processor Intel Core i3 2.3GHz atau lebih
2	RAM minimal 4GB
3	Harddisk Minimal 8GB untuk instalasi software
4	Kamera webcam untuk Service tester minimal 2MP

B. Analisis Kebutuhan Perangkat Keras Pengguna

Kebutuhan perangkat keras *desktop* yang digunakan untuk menjalankan aplikasi *Magic Coloring Book* dengan teknologi *Augmented Reality* dapat dilihat pada Tabel 3.4.

Tabel 3.4 Kebutuhan perangkat keras pengguna

No	Perangkat Keras
1	Layar minimal 5”
2	RAM minimal 1GB
3	Processor minimal 1GHz
4	Kamera minimal 5MP

3.5.2 Analisis Kebutuhan Perangkat Lunak

Analisis perangkat lunak atau *software* merupakah hal yang terpenting dalam mendukung kinerja sebuah sistem. Perangkat lunak digunakan dalam sebuah sistem merupakan perintah-perintah yang diberikan kepada perangkat keras agar dapat saling berinteraksi diantara keduanya. Analisis kebutuhan perangkat lunak ini dibagi menjadi dua bagian, diantaranya seperti :

A. Analisis Kebutuhan Perangkat Lunak Pembangunan Aplikasi

Kebutuhan perangkat lunak yang digunakan untuk membangun aplikasi *Magic Coloring Book* dengan teknologi *Augmented Reality* dapat dilihat pada Tabel 3.5.

Tabel 3.5 Kebutuhan perangkat lunak pembangunan aplikasi

No	Perangkat Lunak
1	Windows 7 Profesional 64-bit atau lebih
2	Unity 3D 4 ver. 4.6.1f1 for Windows
3	Vuforia Android SDK
4	Adobe Photoshop CC 2015
5	StarUML 2.0
6	VGA Driver support for OpenGL rendering

B. Analisis Kebutuhan Perangkat Lunak Pengguna

Kebutuhan perangkat lunak yang dibutuhkan oleh pengguna untuk menjalankan aplikasi *Magic Coloring Book* dengan teknologi Augmented Reality dapat dilihat pada Tabel 3.6.

.

Tabel 3.6 Kebutuhan Perangkat Lunak Pengguna

No	Perangkat Lunak
1	OS Android 4.0.1 atau lebih
2	Graphics Support OpenGL 2.0 keatas

3.5.3 Analisis Kebutuhan Pengguna

Analisis pengguna dimaksudkan untuk mengetahui siapa saja pengguna aplikasi yang akan di bangun ini, maka dari itu analisis terhadap pengguna perlu dilakukan analisis. Pengguna utama aplikasi *Magic Coloring Book* dengan teknologi Augmented Reality ini adalah anak usia 3-6 tahun. Aplikasi ini digunakan dengan pengawasan orang tua. Karakteristik kebutuhan pengguna aplikasi ini dapat dilihat seperti pada Tabel 3.7.

Tabel 3.7 Analisis kebutuhan pengguna

Pengguna	Hak Akses	Tingkat Pendidikan	Pengalaman
Semua usia	Menjalankan fitu-fitur yang ada pada aplikasi	Tidak dibatasi	Dapat menggunakan perangkat ponsel pintar (*smartphone*)

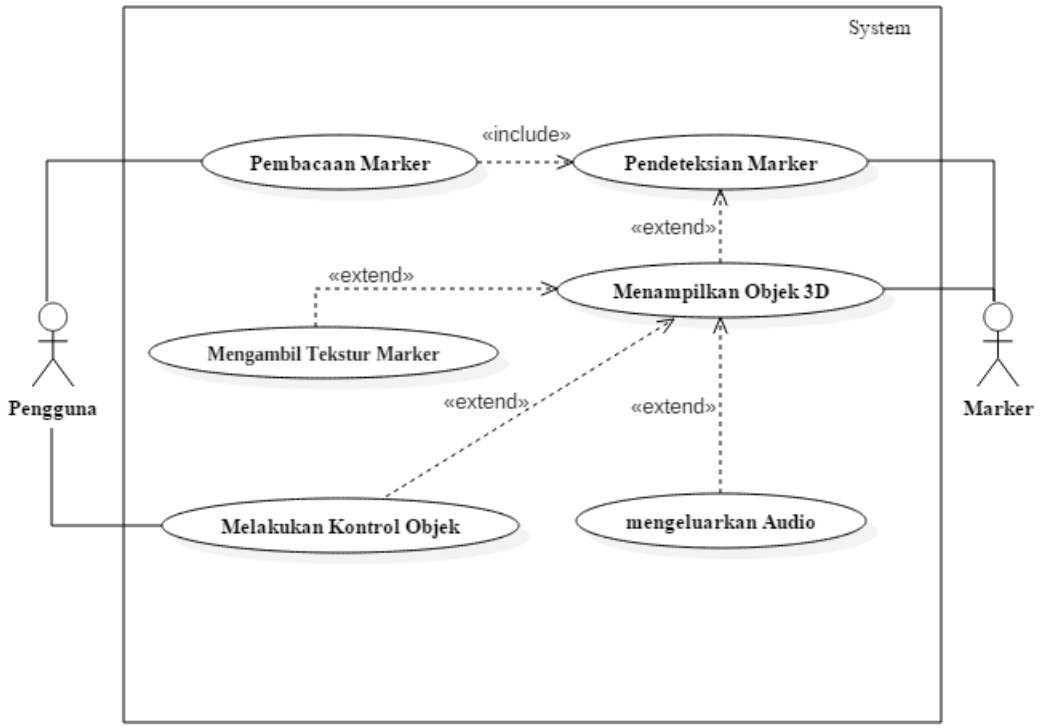
3.6 Analisis Kebutuhan Fungsional

Analisis kebutuhan fungsional ini dimodelkan dengan menggunakan UML (*Unified Modeling Language*). Dimana tahap-tahap perancangan yang dilakukan dalam membangun aplikasi *magic coloring book* dengan teknologi *augmented reality* antara lain *Use Case Diagram*, *Class Diagram*, *Sequence Diagram*, dan *Activity Diagram.*

3.6.1 Use Case Diagram

Use case Diagram merupakan konstruksi untuk mendeskripsikan hubungan-hubungan yang terjadi antar aktor dengan aktivitas yang terdapat pada aplikasi. Sasaran pemodelan *use case* diantaranya adalah mendefinisikan kebutuhan fungsional dan operasional aplikasi dengan mendefinisikan skenario penggunaan aplikasi yang akan dibangun.

Berikut ini adalah perancangan proses-proses yang terdapat pada aplikasi *magic coloring book* dengan teknologi *augmented reality*, yang digambarkan dengan *Use Case Diagram* yang dapat dilihat pada Gambar 3.16.



Gambar 3.16 Usecase Diagram Aplikasi

3.6.1.1 Definisi Aktor

Aktor adalah abstraksi dari orang dan sistem yang lain yang mengaktifkan fungsi dari target sistem. Berikut adalah aktor yang berperan dalam menjalankan sistem yang dibangun. Dapat dilihat pada Tabel 3.8.

Tabel 3.8 Aktor dalam *use case*

No	Actor	Deskripsi
A-01	Pengguna	Merupakan aktor atau pengguna dari aplikasi yang dibangun yang akan mengakses dan menggunakan fitur-fitur yang tersedia pada aplikasi
A-02	Marker	Sebagai inputan dalam proses augmented reality

3.6.1.2 Definisi *Use Case*

Diagram *use case* merupakan konstruksi untuk mendeskripsikan hubungan-hubungan yang terjadi antar aktor dengan aktivitas yang terdapat pada sistem. Tujuan pemodelan *use case* diantaranya adalah mendefinisikan kebutuhan

fungsional dan operasi sistem. Definisi dari setiap *use case* yang ada pada sistem dapat dilihat pada Tabel 3.9.

Tabel 3.9 Definisi *use case*

No	Nama Use Case	Deskripsi
UC-01	Pembacaan Marker	Fungsionalitas untuk membaca *marker*
UC-02	Pendeteksian Marker	Fungsionalitas untuk mengenali *marker*
UC-03	Menampilkan Objek 3D	Fungsionalitas untuk menampilkan objek 3D
UC-04	Mengeluarkan Audio	Fungsionalitas untuk mengeluarkan objek suara
UC-05	Mengambil Tekstur Marker	Fungsionalitas untuk mengambil tekstur warna dari marker yang dideteksi
UC-06	Melakukan Kontrol Objek	Fungsionalitas untuk mengontrol objek 3D yang ditampilkan

3.6.2 Skenario *Use Case*

Skenario *Use Case* berisi mengenai skenario (*flow of event*) untuk beberapa *use case* utama, yang menggambarkan urutan interaksi *actor* dengan *use case* tersebut dari awal sampai akhir.

3.6.2.1 Skenario Use Case Pembacaan Marker

Skenario yang berkaitan dengan *use case* pembacaan marker dapat dilihat seperti pada Tabel 3.10.

Tabel 3.10 Skenario Use Case Inisialisasi Kamera

Nama Use Case	Pembacaan Marker	
Persyaratan Terkait	-	
Tujuan Dalam konteks	Aplikasi dapat menampilkan objek 3D beserta audio	
Kondisi Awal	Kamera belum terdeteksi	
Kondisi Akhir Berhasil	Aplikasi dapat mengenali marker dan dapat menampilkan objek 3D beserta audio	
Kondisi Akhir Gagal	Aplikasi tidak dapat menginisialisasi marker	
Aktor Utama	Pengguna	
Aktor Kedua	Marker	
Include	Pendeteksian Marker	
Extend	-	
Trigger	Pengguna memilih menu augmented reality	
Aliran Utama	**Langkah**	**Aksi**
	1.	Memilih menu deteksi *augmented reality*
	2.	Pengguna mengarahkan marker pada kamera
	3. *include::pendeteksian marker*	Aplikasi mendeteksi marker
	4.	Aplikasi me-*render* objek 3D
	5.	Aplikasi menampilkan objek 3D sesuai *marker* yang terdeteksi
Perluasan	**Langkah**	**Aksi Percabangan**
	4.1.	Aplikasi tidak dapat mengenali *marker* yang terdeteksi
	5.1	Aplikasi tidak dapat merender objek 3D sesuai dengan *marker*

3.6.2.2 Skenario Use Case Pendeteksian Marker

Skenario yang berkaitan dengan *use case* pendeteksian marker dapat dilihat seperti pada Tabel 3.11.

Tabel 3.11 Skenario Use Case Pendeteksian Marker

Nama Use Case	Pendeteksian Marker	
Persyaratan Terkait	-	
Tujuan Dalam konteks	Aplikasi dapat mengenali marker yang terdeteksi	
Kondisi Awal	Marker belum terdeteksi	
Kondisi Akhir Berhasil	Aplikasi dapat mengenali marker	
Kondisi Akhir Gagal	Aplikasi tidak dapat mengenali marker	
Aktor Utama	Pengguna	
Aktor Kedua	Marker	
Include	-	
Extend	Menampilkan Objek 3D	
Trigger	Kamera dapat mendeteksi keberadaan marker	
Aliran Utama	**Langkah**	**Aksi**
	1.	Pengguna 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*
Perluasan	**Langkah**	**Aksi Percabangan**
	3.1.	Aplikasi tidak mengidentifikasi marker
	6.1	Aplikasi tidak dapat mendeteksi marker

3.6.2.3 Skenario Use Case Menampilkan Objek 3D

Skenario yang berkaitan dengan *use case* menampilkan objek 3D dapat dilihat seperti pada Tabel 3.12.

Tabel 3.12 Skenario Use Case Menampilkan Objek 3D

Nama Use Case	Menampilkan Objek 3D	
Persyaratan Terkait	-	
Tujuan Dalam konteks	Aplikasi menampilkan objek 3D	
Kondisi Awal	Marker yang terdeteksi belum dapat dikenali	
Kondisi Akhir Berhasil	Aplikasi menampilkan objek 3D sesuai marker yang dideteksi tepat di atas marker	
Kondisi Akhir Gagal	Aplikasi tidak dapat menampilkan objek 3D	
Aktor Utama	Pengguna	
Aktor Kedua	Marker	
Include	-	
Extend	Mengeluarkan Audio, Mengambil Tekstur Marker, Melakukan Kontrol Objek	
Trigger	Marker dapat dikenali dengan baik dan sesuai dengan pola marker yang ada	
Aliran Utama	**Langkah**	**Aksi**
	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
Perluasan	**Langkah**	**Aksi Percabangan**
	3.1.	Aplikasi tidak dapat menemukan pola marker yang sesuai
	3.2.	Aplikasi tidak merender objek 3D
	3.3.	Aplikasi tidak menampilkan objek 3D
	5.1.	Aplikasi mengeluarkan audio dari objek 3D
	5.2.	Aplikasi menampilkan objek 3D dengan tekstur yang sesuai dengan marker
	5.3	Aplikasi menampilkan kontrol objek pada layar aplikasi

3.6.2.4 Skenario Use Case Mengeluarkan Audio

Skenario yang berkaitan dengan *use case* mengeluarkan audio dapat dilihat seperti pada Tabel *3.13*.

Tabel 3.13 Skenario Use Case Mengeluarkan Audio

Nama Use Case	Mengeluarkan Audio	
Persyaratan Terkait	-	
Tujuan Dalam konteks	Aplikasi mengeluarkan audio dari objek yang ditampilkan	
Kondisi Awal	Objek 3D sudah ditampilkan	
Kondisi Akhir Berhasil	Aplikasi dapat mengeluarkan audio dari objek 3D yang ditampilkan	
Kondisi Akhir Gagal	Aplikasi tidak dapat mengeluarkan audio	
Aktor Utama	Pengguna	
Aktor Kedua	Marker	
Trigger	Objek 3D telah ditampilkan dan pengguna menekan tombol suara objek	
Aliran Utama	**Langkah**	**Aksi**
	1.	Pengguna menekan tombol suara objek
	2.	Aplikasi mengeluarkan audio objek
Perluasan	**Langkah**	**Aksi Percabangan**
	2.1.	Aplikasi tidak dapat mengeluarkan suara objek

3.6.2.5 Skenario Use Case Mengambil Tekstur Marker

Skenario yang berkaitan dengan *use case* mengambil tekstur marker dapat dilihat seperti pada Tabel *3.14*.

Tabel 3.14 Skenario Use Case Mengambil Tekstur Marker

Nama Use Case	Mengambil Tekstur Marker	
Persyaratan Terkait	-	
Tujuan Dalam konteks	Aplikasi dapat mengambil tekstur warna dari marker yang dideteksi	
Kondisi Awal	Objek 3D sudah ditampilkan	
Kondisi Akhir Berhasil	Aplikasi dapat mengambil tekstur warna dari marker yang dideteksi kemudian mengaplikasikannya pada objek 3D yang ditampilkan	
Kondisi Akhir Gagal	Aplikasi tidak dapat mengambil tekstur warna dari marker yang dideteksi	
Aktor Utama	Sistem	
Trigger	Marker terdeteksi	
Aliran Utama	**Langkah**	**Aksi**
	1.	Aplikasi mendeteksi marker
	2.	Aplikasi meng-capture marker yang terdeteksi
	3.	Aplikasi mengubah citra hasil capture menjadi tekstur 2D
	4.	Aplikasi mengaplikasikan tekstur 2D Yan didapat terhadap objek 3D yang akan ditampilkan
Perluasan	**Langkah**	**Aksi Percabangan**
	1.1.	Aplikasi tidak dapat mendeteksi marker
	2.1	Aplikasi tidak dapat meng-capture marker yang terdeteksi

3.6.2.6 Skenario Use Case Menampilkan Kontrol Objek

Skenario yang berkaitan dengan *use case* menampilkan kontrol objek dapat dilihat seperti pada Tabel *3.15*.

Tabel 3.15 Skenario Use Case Menampilkan Kontrol Objek

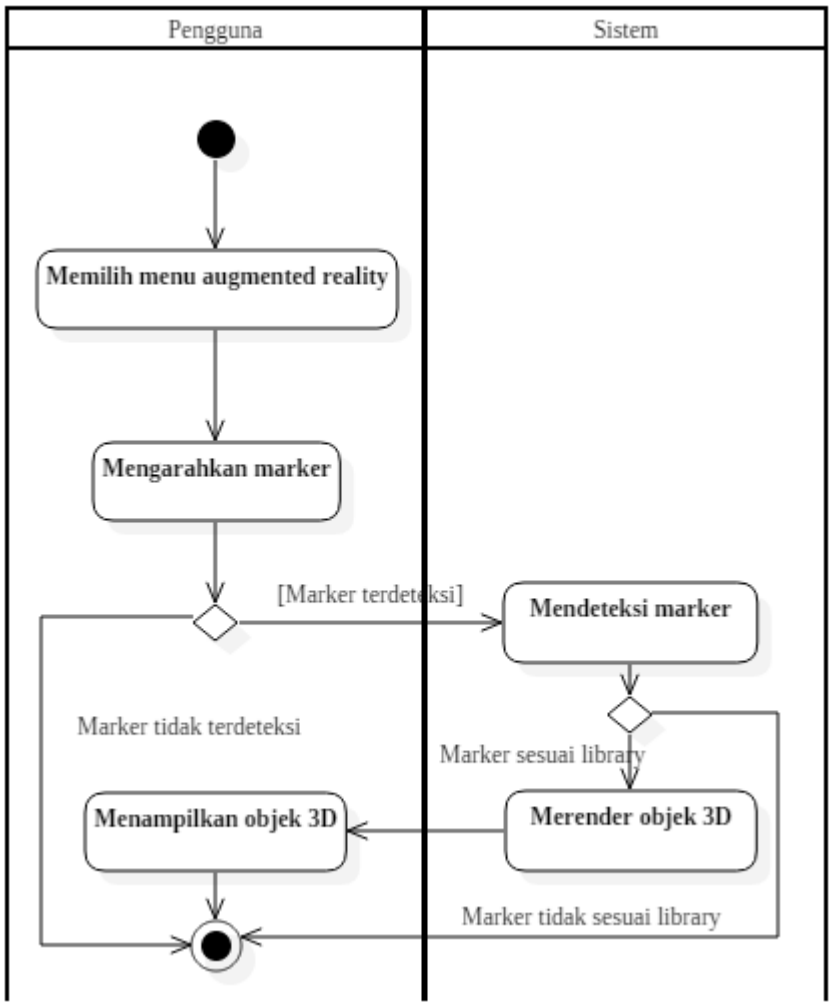
Nama Use Case	Menampilkan Kontrol Objek	
Persyaratan Terkait	-	
Tujuan Dalam konteks	Aplikasi menampilkan tombol kontrol objek	
Kondisi Awal	Objek 3D sudah ditampilkan	
Kondisi Akhir Berhasil	Aplikasi dapat menampilkan tombol untuk mengontrol objek 3D yang ditampilkan	
Kondisi Akhir Gagal	Aplikasi tidak dapat menampilkan tombol kontrol objek	
Aktor Utama	Sistem	
Aktor Kedua	Pengguna	
Trigger	Objek 3D telah ditampilkan	
Aliran Utama	**Langkah**	**Aksi**
	1.	Aplikasi menampilkan objek 3D
	2.	Aplikasi menampilkan tombol kontrol objek
	3.	Pengguna menekan tombol kontrol untuk mengontrol objek yang ditampilkan
Perluasan	**Langkah**	**Aksi Percabangan**
	2.1.	Aplikasi tidak dapat menampilkan tombol kontrol objek

3.6.3 Activity Diagram

Activity diagram menggambarkan berbagai aliran aktivitas dalam sistem yang dirancang, bagaimana masing-masing fungsionalitas bekerja, dan bagaimana suatu fungsional berakhir. *Activity diagram* memodelkan *event-event* yang terjadi pada use case.

3.6.3.1 Activity Diagram Pembacaan Marker

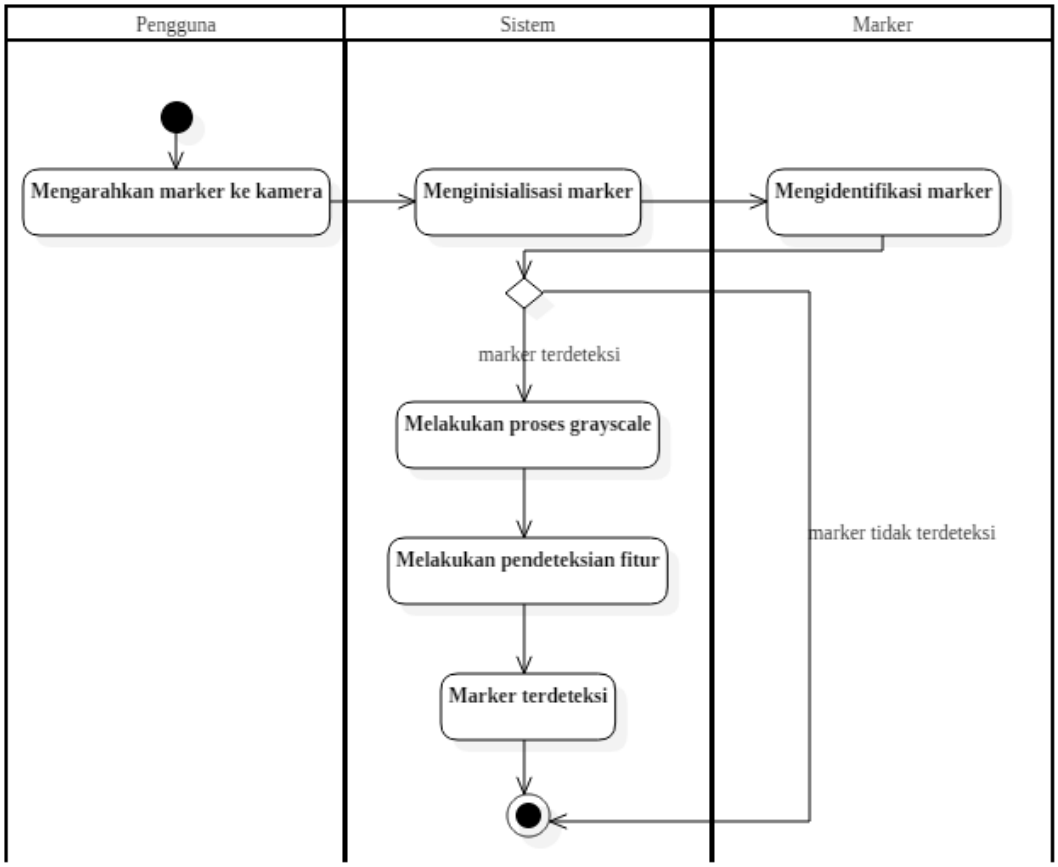
Merupakan aktivitas dari aplikasi untuk membaca marker. *Activity diagram* terkait dengan skenario *use case* pembacaan marker dapat dilihat seperti pada Gambar 3.17.



Gambar 3.17 Activity Diagram Pembacaan Marker

3.6.3.2 Activity Diagram Pendeteksian Marker

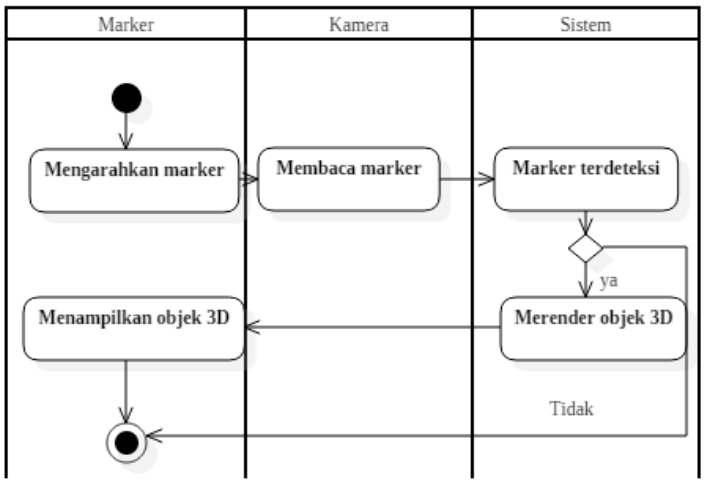
Merupakan sub-aktifitas dari *use case* pendeteksian marker. *Activity diagram* dari pendeteksian marker dapat dilihat pada Gambar 3.18.



Gambar 3.18 Activity Diagram Pendeteksian Marker

3.6.3.3 Activity Diagram Menampilkan Objek 3D

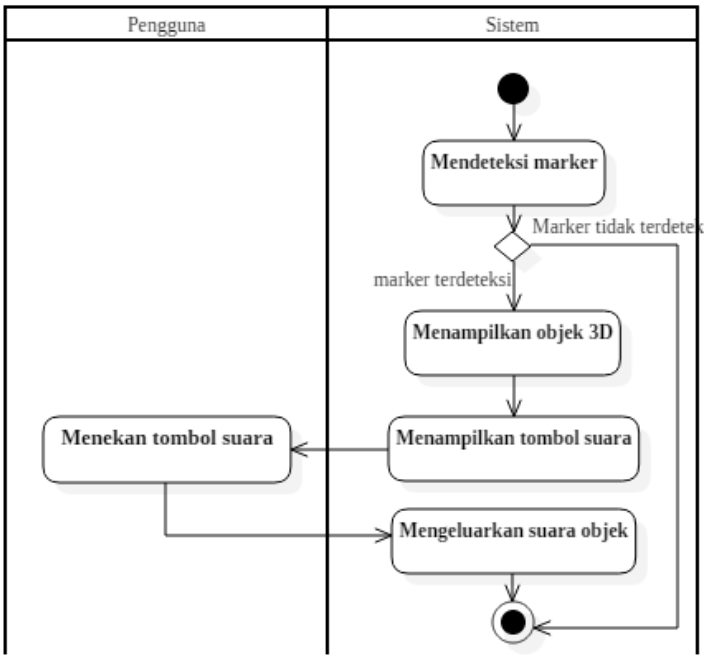
Merupakan sub-aktifitas dari *use case* menampilkan objek 3D. *Activity diagram* dari menampilkan objek 3D dapat dilihat pada Gambar 3.19.



Gambar 3.19 Activity Diagram Menampilkan Objek 3D

3.6.3.4 Activity Diagram Mengeluarkan Audio

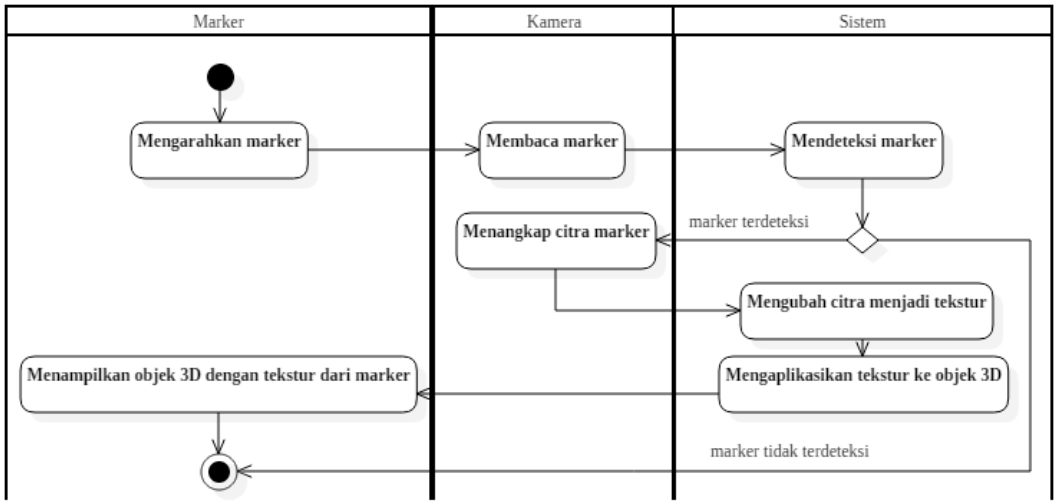
Merupakan sub-aktifitas dari *use case* mengeluarkan audio. *Activity diagram* dari mengeluarkan audio dapat dilihat pada Gambar 3.20.



Gambar 3.20 Activity Diagram Mengeluarkan Audio

3.6.3.5 Activity Diagram Mengambil Tekstur Marker

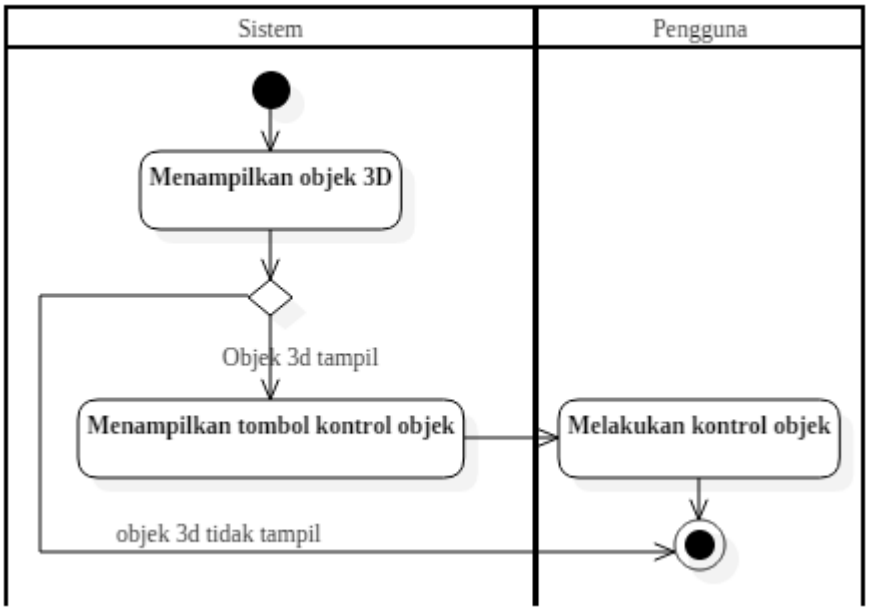
Merupakan sub-aktifitas dari *use case* mengambil tekstur marker. *Activity diagram* dari mengambil tekstur marker dapat dilihat pada Gambar 3.21.



Gambar 3.21 Activity Diagram Mengambil Tekstur Marker

3.6.3.6 Activity Diagram Menampilkan Kontrol Objek

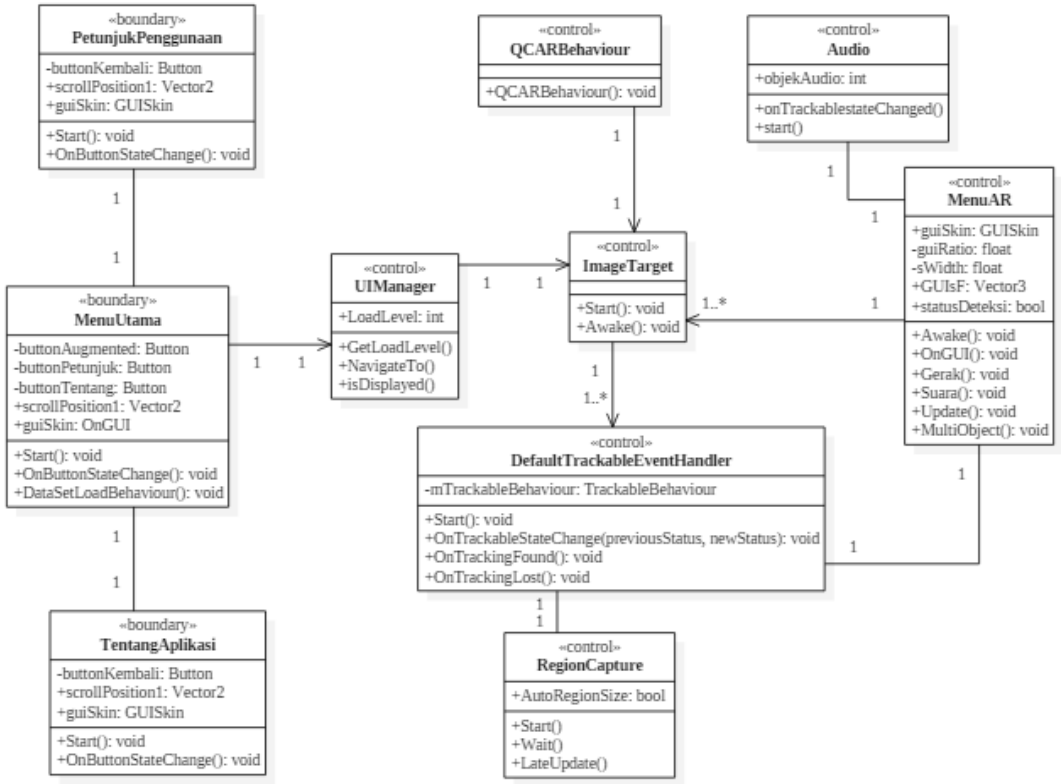
Merupakan sub-aktifitas dari *use case* menampilkan kontrol objek. *Activity diagram* dari menampilkan kontrol objek dapat dilihat pada Gambar 3.22.



Gambar 3.22 Activity Diagram Menampilkan Kontrol Objek

3.6.4 Class Diagram

Class Diagram menggambarkan struktur dan hubungan antar objek-objek yang ada pada sistem. Struktur itu meliputi atribut-atribut dan metode-metode yang ada pada masing-masing kelas. *Class diagram* dari aplikasi "*Magic Coloring Book*" dengan teknologi *Augmented Reality* ini dapat dilihat seperti pada Gambar 3.23.



Gambar 3.23 *Class Diagram*

Tabel deskripsi kelas dari *class diagram* pada Gambar 3.23 dapat dilihat pada Tabel 3.16.

Tabel 3.16 Deskripsi Kelas

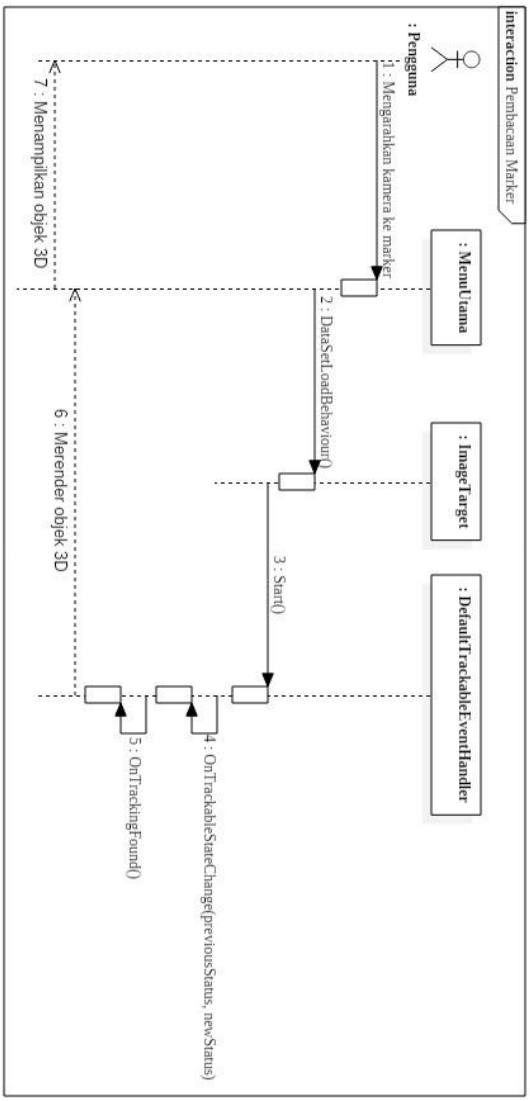
No	Nama Kelas	Tipe Kelas
1	MenuUtama	Boundary
2	TentangAplikasi	Boundary
3	PetunjukPenggunaan	Boundary
4	ImageTarger	Control
5	QCARBehaviour	Control
6	DefaultTrackableEventHandler	Control
7	UIManager	Control
8	RegionCapture	Control
9	Audio	Control
10	MenuAR	Control

3.6.5 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.

3.6.5.1 Sequence Diagram Pembacaan Marker

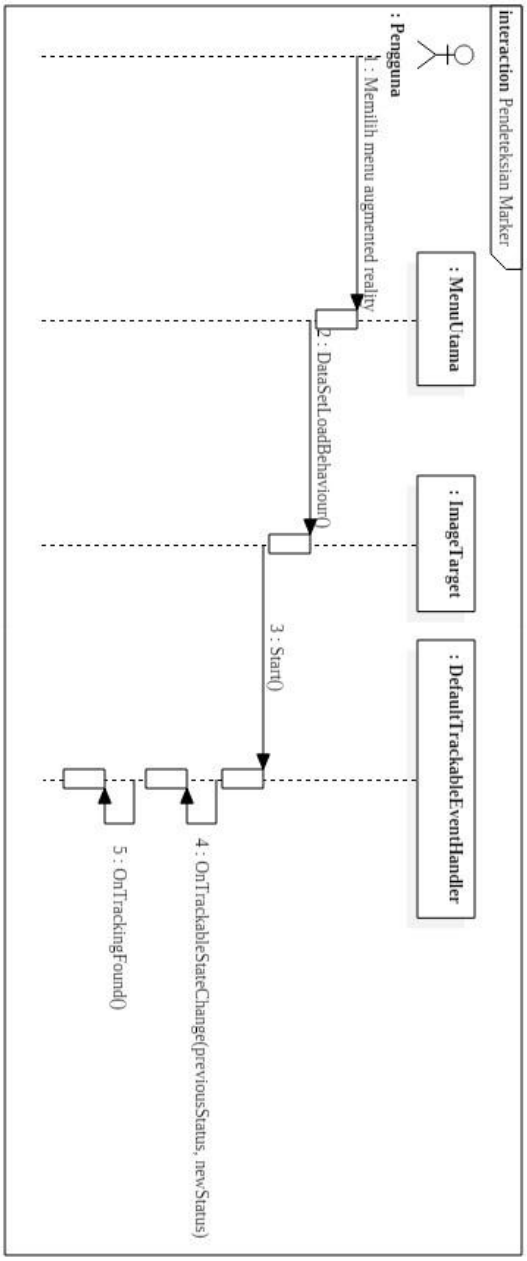
Sequence diagram untuk proses pembacaan *marker* dapat dilihat pada Gambar 3.24.



Gambar 3.24 *Sequence Diagram* Pembacaan Marker

3.6.5.2 Sequence Diagram Pendeteksian Marker

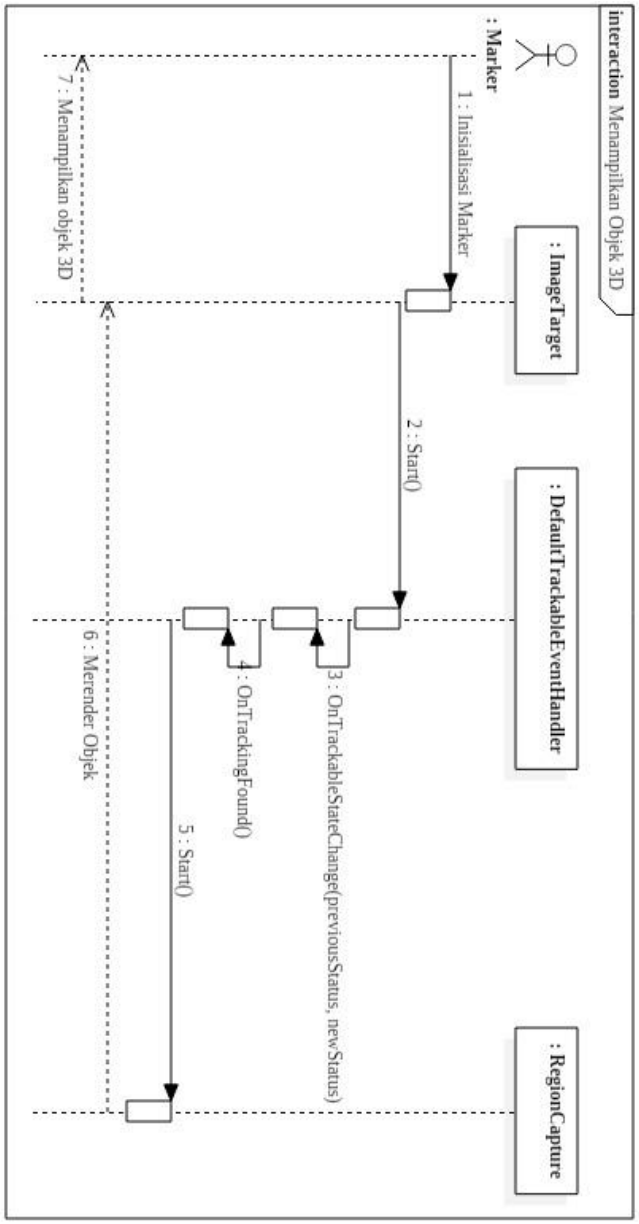
Sequence diagram untuk proses pendeteksian *marker* dapat dilihat pada Gambar 3.25.



Gambar 3.25 *Sequence Diagram* Pendeteksian Marker

3.6.5.3 Sequence Diagram Menampilkan Objek 3D

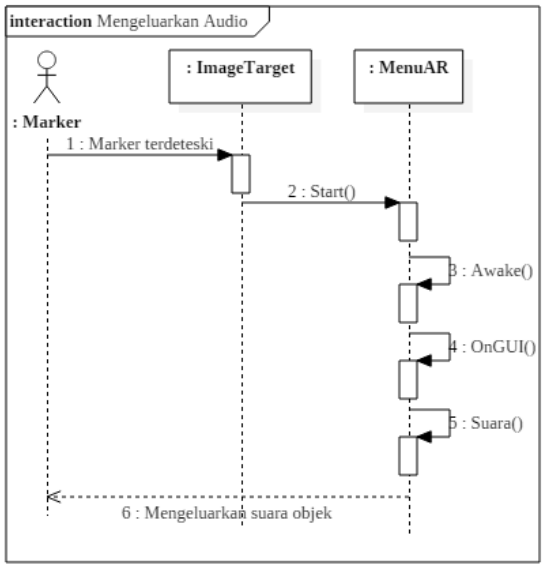
Sequence diagram untuk menampilkan objek 3D dapat dilihat pada Gambar 3.26.



Gambar 3.26 *Sequence Diagram* Menampilkan Objek 3D

3.6.5.4 Sequence Diagram Mengeluarkan Audio

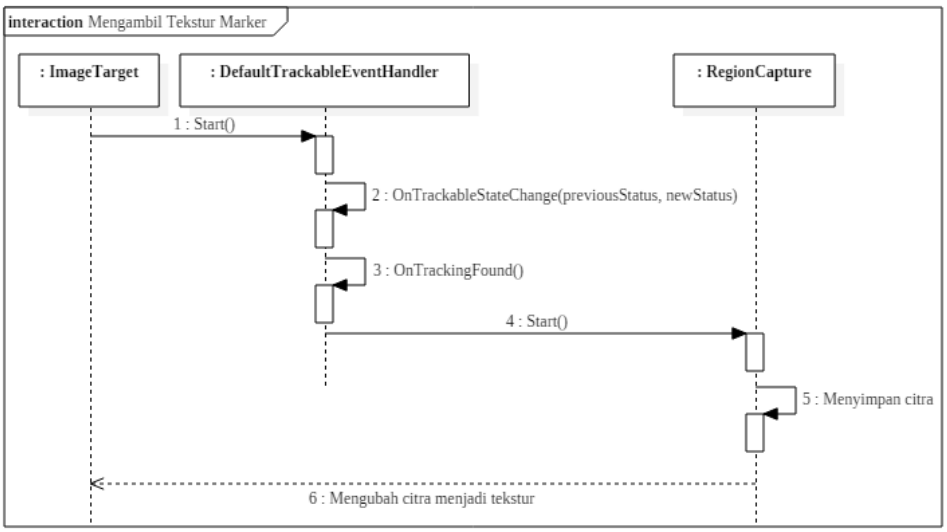
Sequence diagram untuk proses mengeluarkan audio dapat dilihat pada Gambar 3.27.



Gambar 3.27 *Sequence Diagram* Mengeluarkan Audio

3.6.5.5 Sequence Diagram Mengambil Tekstur Marker

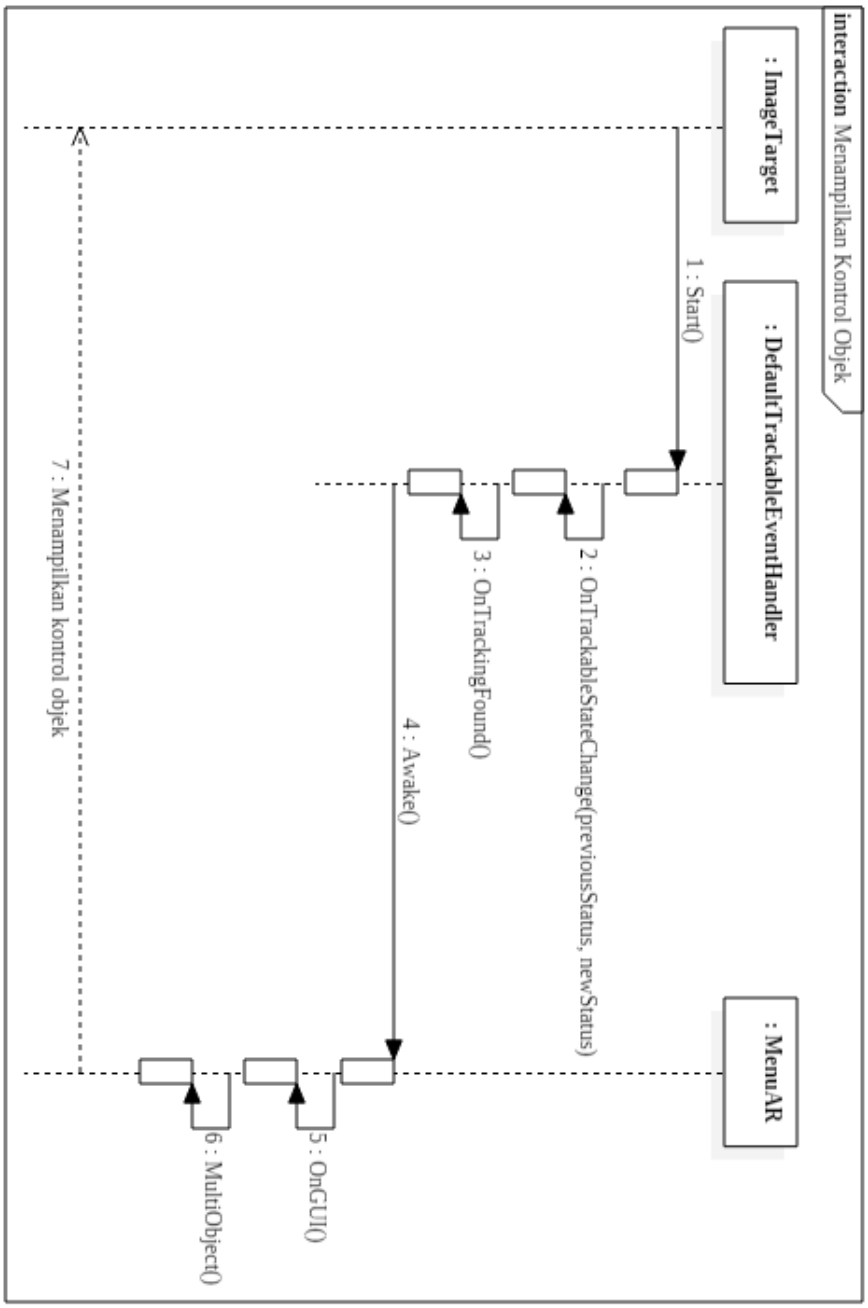
Sequence diagram untuk mengambil tekstur marker dapat dilihat pada Gambar 3.28.



Gambar 3.28 *Sequence Diagram* Mengambil Tekstur Marker

3.6.5.6 Sequence Diagram Menampilkan Kontrol Objek

Sequence diagram untuk menampilkan kontrol objek dapat dilihat pada Gambar 3.29.



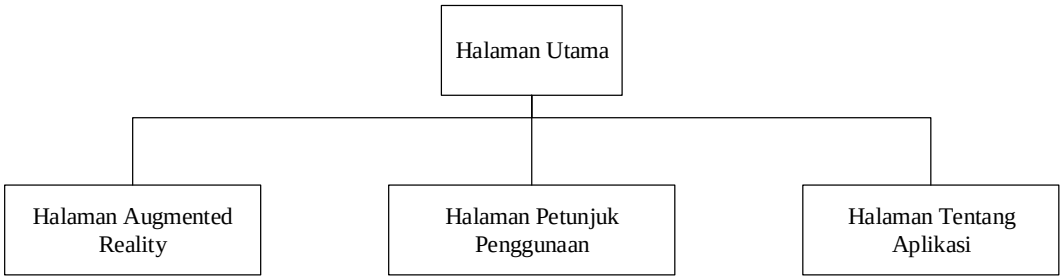
Gambar 3.29 *Sequence Diagram* Menampilkan Kontrol Objek

3.7 Perancangan Sistem

Perancangan sistem bertujuan untuk memberikan gambaran aplikasi yang dibuat. Perancangan disini terdiri dari perancangan struktur menu, perancangan antarmuka, jaringan semantik, serta perancangan *method*.

3.7.1 Perancangan Struktur Menu

Perancangan struktur *menu* merupakan gambaran jalur pemakaian dari aplikasi sehingga aplikasi menjadi mudah dimengerti serta mudah digunakan dan juga saat menggunakan aplikasi, pengguna tidak mengalami banyak kesulitan dalam memilih *menu-menu* yang diinginkan. Perancangan struktur menu pada aplikasi ini dapat dilihat pada Gambar 3.30.



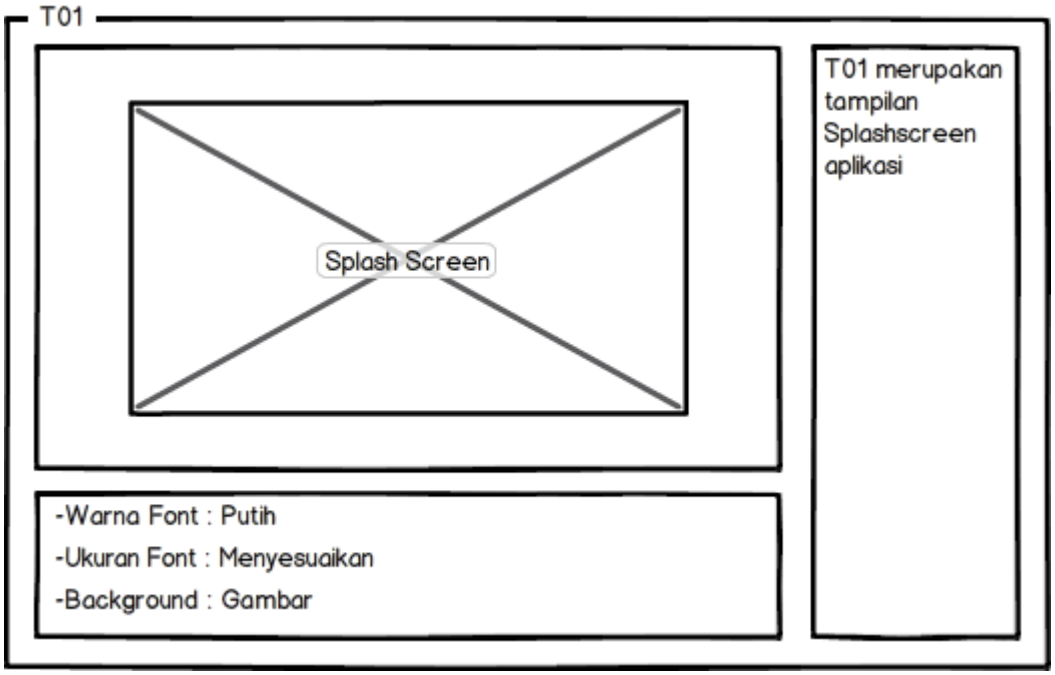
Gambar 3.30 Perancangan Struktur Menu

3.7.2 Perancangan Antarmuka

Perancangan antarmuka merupakan rancangan tampilan dari aplikasi. Perancangan antarmuka aplikasi ini terdiri dari *splash screen*, Halaman Utama, Halaman *Augmented Reality*, Halaman petunjuk penggunaan, Halaman tentang aplikasi, dan Halaman AR.

3.7.2.1 Perancangan Antarmuka Splash Screen

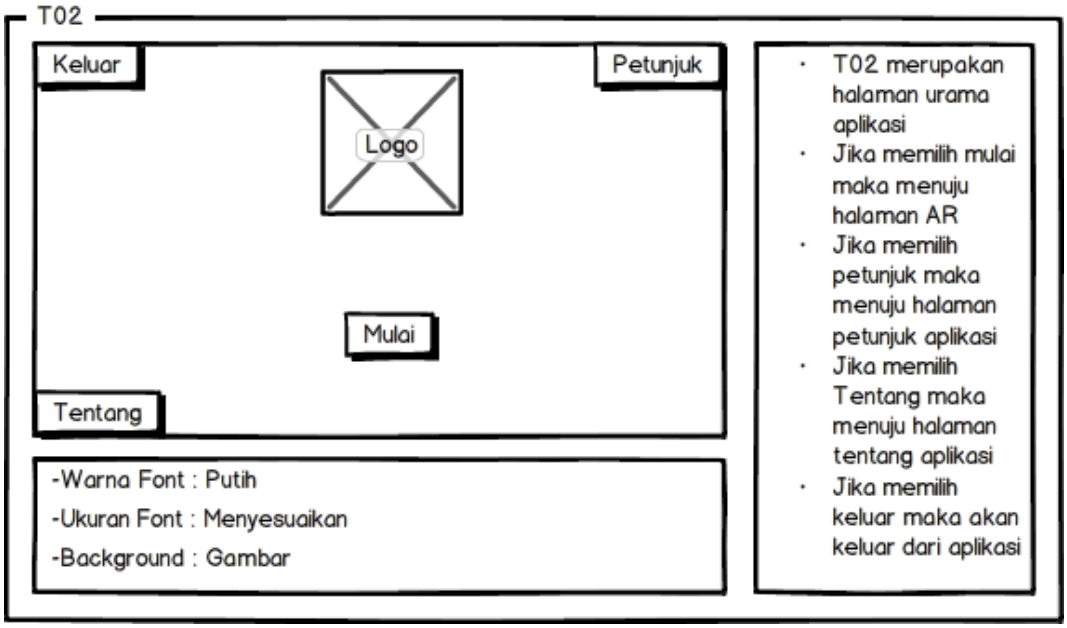
Perancangan antarmuka splash screen aplikasi ini dapat dilihat seperti pada Gambar 3.31.



Gambar 3.31 Perancangan Antarmuka Splashscreen

3.7.2.2 Perancangan Antarmuka Halaman Utama

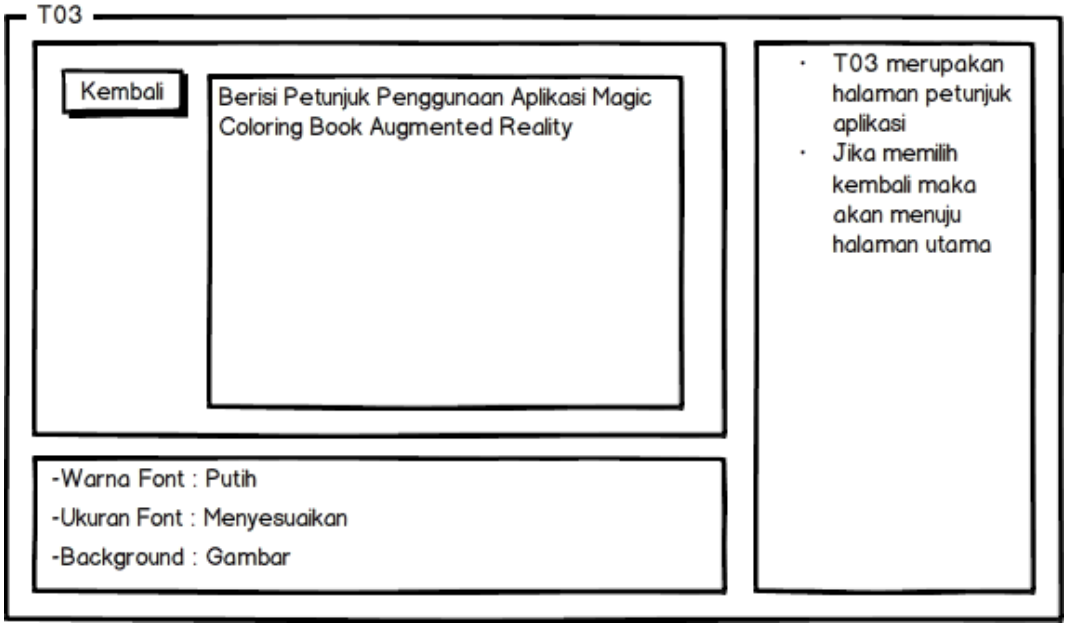
Perancangan antarmuka halaman utama aplikasi ini dapat dilihat seperti pada Gambar 3.32.



Gambar 3.32 Perancangan Antarmuka Halaman Utama Aplikasi

3.7.2.3 Perancangan Antarmuka Halaman Petunjuk Penggunaan

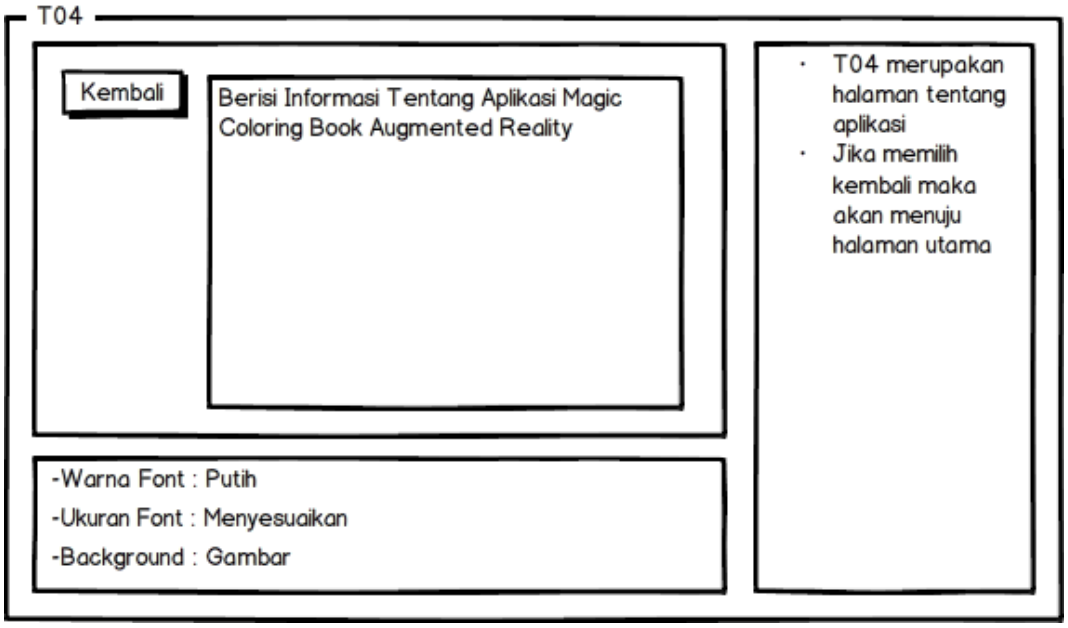
Perancangan antarmuka halaman petunjuk penggunaan aplikasi ini dapat dilihat seperti pada Gambar 3.33.



Gambar 3.33 Perancangan Antarmuka Petunjuk Penggunaan Aplikasi

3.7.2.4 Perancangan Antarmuka Halaman Tentang Aplikasi

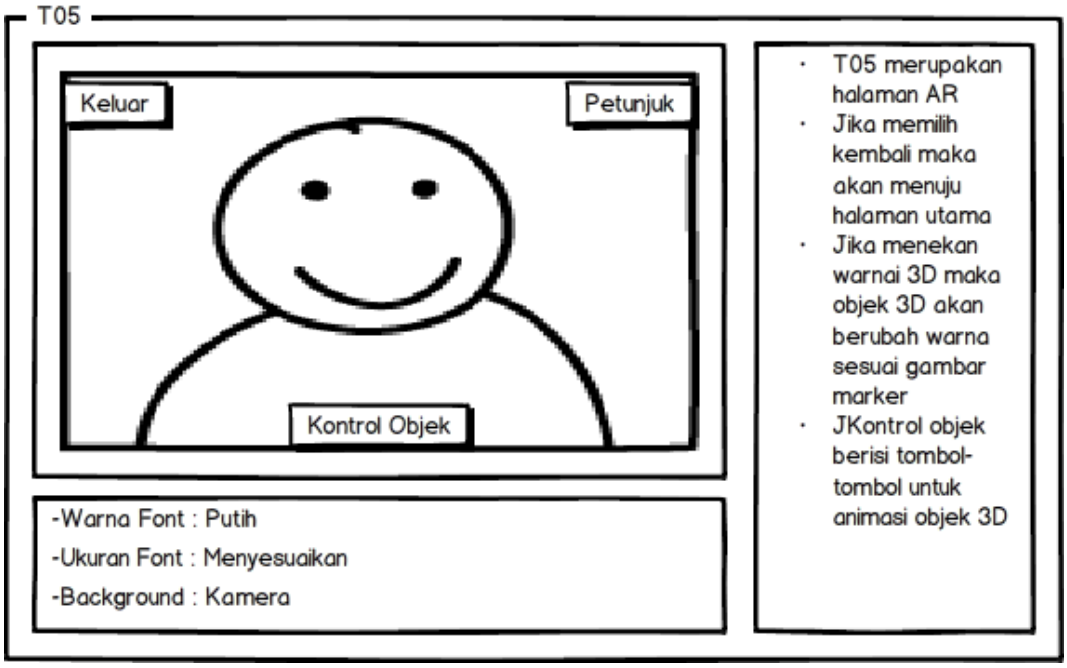
Perancangan antarmuka halaman tentang aplikasi ini dapat dilihat seperti pada Gambar 3.34.



Gambar 3.34 Perancangan Antarmuka Tentang Aplikasi

3.7.2.5 Perancangan Antarmuka Halaman AR

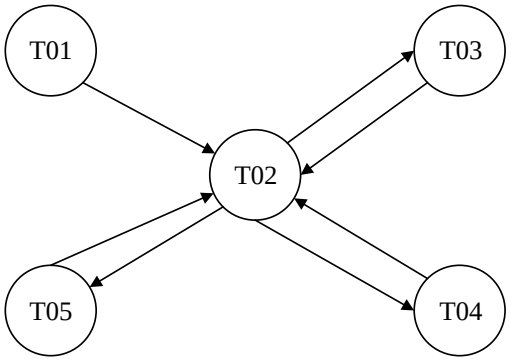
Perancangan antarmuka halaman AR aplikasi ini dapat dilihat seperti pada Gambar 3.35.



Gambar 3.35 Perancangan Antarmuka Halaman AR

3.7.3 Perancangan Semantik

Perancangan jaringan semantik dari aplikasi ini dapat dilihat seperti pada



Gambar 3.36 Jaringan Semantik Aplikasi

Keterangan:

T01 : Tampilan Splashscreen

T02 : Tampilan Halaman Utama

T03 : Tampilan Halaman Petunjuk Penggunaan

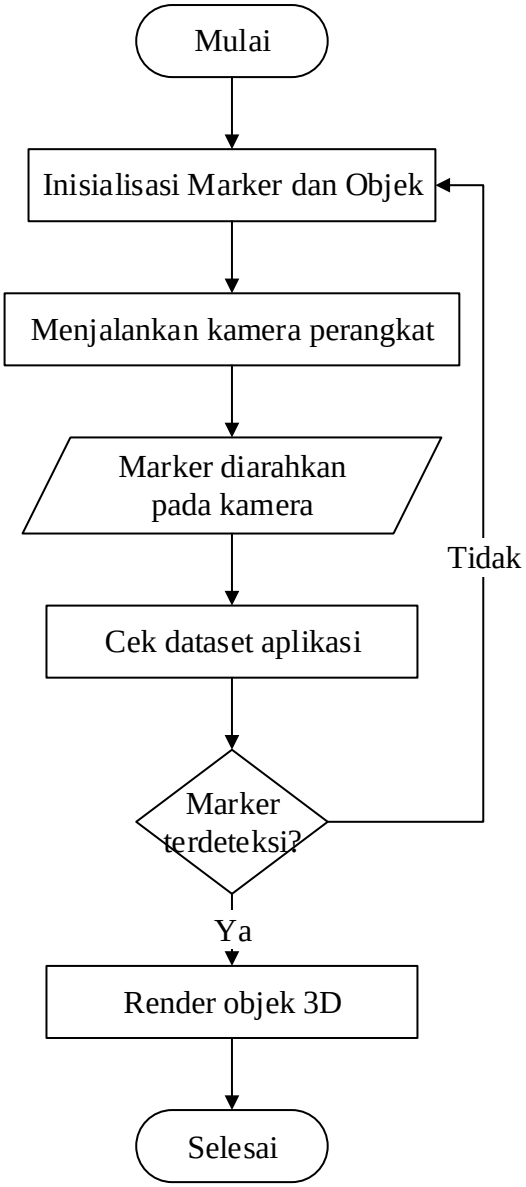
T04 : Tampilan Halaman Tentang Aplikasi

T05 : Tampilan Halaman AR

3.7.4 Perancangan Method

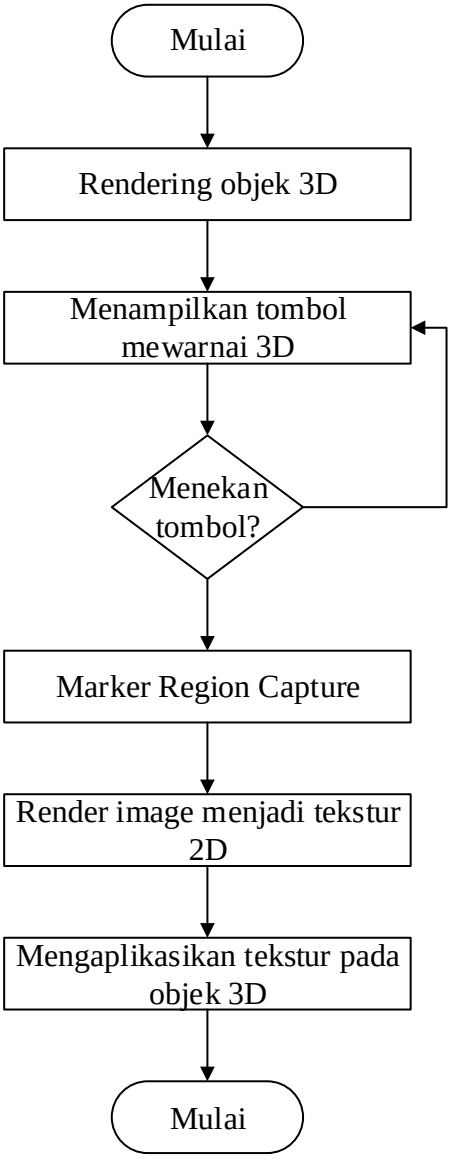
Perancangan *method* berfungsi untuk mendeskripsikan *method* yang berada dalam aplikasi. *Method* yang terdapat pada aplikasi yang akan dibangun yaitu *flowchart* menampilkan objek rumah dan *flowchart* mengontrol objek. *Method* atau *flowchart* tersebut dapat dilihat sebagai berikut:

1. *Flowchart* Menampilkan Objek 3D



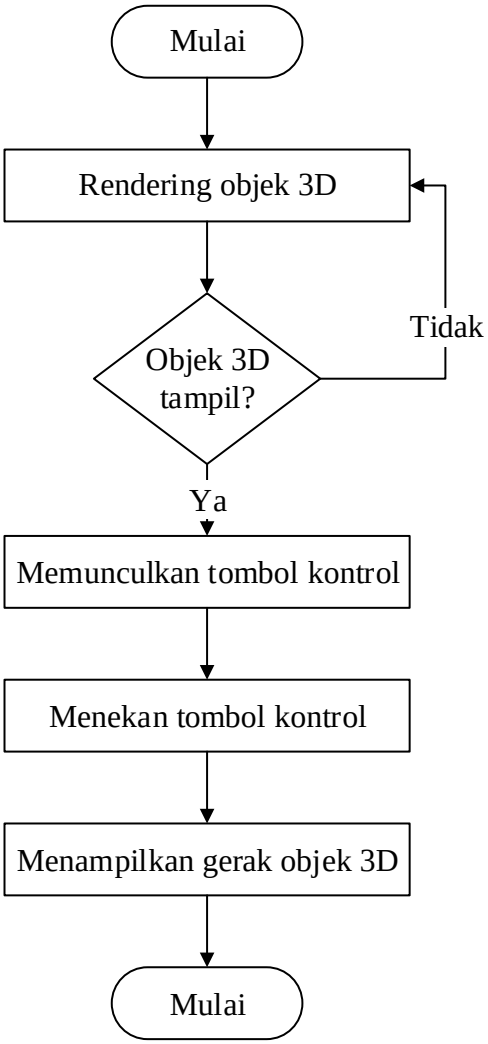
Gambar 3.37 *Flowchart* Menampilkan Objek 3D

2. *Flowchart* Merender Tekstur Marker



Gambar 3.38 *Flowchart* Merender Tekstur Maker

3. *Flowchart* Mengontrol Objek



Gambar 3.39 *Flowchart* Mengontrol Objek