BAB 3
ANALISIS DAN PERANCANGAN

3.1 Analisis Masalah

Setiap siswa memiliki daya tangkap terhadap materi pelajaran yang berbeda-beda, tidak terkecuali materi sistem pernafasan pada manusia yang cukup rumit dan sulit untuk dipelajari dan sulit untuk mendapatkan informasi yang lengkap tentang organ-organ penyususnnya. Alat peraga yang kurang memadai menjadi masalah yang selalu muncul di setiap sekolah untuk siswa dapat berinteraksi dengan apa yang akan mereka pelajari.

Analisis masalah sendiri bertujuan untuk menggambarkan suatu masalah dalam penulisan skripsi tentang multimedia presentasi pembelajaran berbasis *augmented reality* untuk sistem pernafasan pada manusia yang dapat meningkatkan daya pemahaman siswa mengenai materi sistem pernafasan dengan memanfaatkan teknologi *augmented reality*.

3.1.1 Analisis Sistem

Analisis merupakan penguraian dari suatu sistem yang utuh ke dalam bagian-bagian komponen dengan maksud untuk mengidentifikasi dan mengevaluasi permasalahan, sehingga diperoleh solusi. Analisis merupakan tahapan yang paling penting, karena kesalahan dalam tahap ini akan menyebabkan kesalahan di tahap selanjutnya.

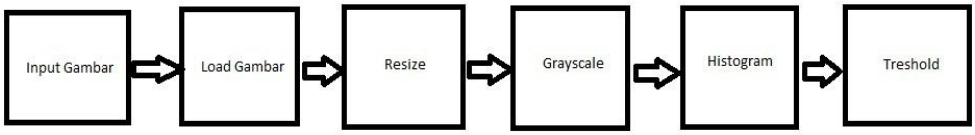
3.1.2 Analisis Pembuatan Marker dengan Aruco Library

Marker adalah real *environment* berbentuk objek nyata yang akan menghasilkan *virtual reality*, marker ini digunakan sebagai tempat *Augmented Reality* muncul [10], berikut ini tahap-tahap bagaimana *Openspace 3D* melakukan konversi gambar menjadi marker:

1. Inisialisasi Gambar
2. *Tracking Markerless*

3.1.2.1 Inisialisasi Gambar

Aruco Library merupakan *plug-in library* pada *openspace.* Pada tahap ini ditentukan proses dimana *Aruco Library* bekerja untuk mengambil gambar sebagai data masukan yang akan diproses. Seperti terlihat pada gambar 3.1.



Gambar 3.1 Alur sistem proses inisialisasi gambar

1. Input Gambar

Pada saat aplikasi berjalan, kamera melakukan proses pengambilan gambar dalam format *.jpg.

```
/**Set the matching type. */
aRmatcher.setMatchingType(ARmatcher.IMAGE_MATCHER);/*proses
memasukkan gambar*/
```

Gambar masukan berbentuk format *.jpg akan diproses oleh *Aruco Library.*

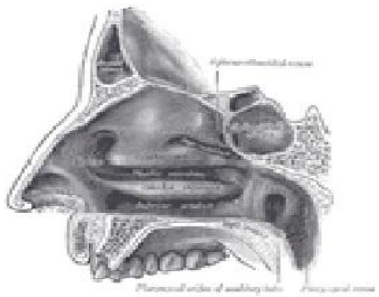
2. Load Gambar

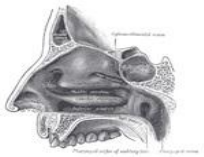
Proses inisialisasi berawal dari pengambilan gambar secara otomatis menggunakan kamera *webcam.* Sistem berjalan dengan didukung oleh *Aruco Library* dari *OpenSpace.*

```
surfaceView=(SurfaceView)findViewById(R.id.surfaceView);
holder=surfaceView.getHolder();
holder.addCallback(this);/*proses memanggil gambar*/
holder.setType(SurfaceHolder.SURFACE_TYPE_PUSH_BUFFERS);
```

3. Resize

Gambar yang dijadikan sebagai *markerless* akan di-*resize* terlebih dahulu. Sesuai ketentuan *Aruco Library*, gambar akan di-*resize* mejadi ukuran 250x250. Seperti terlihat pada gambar 3.2.





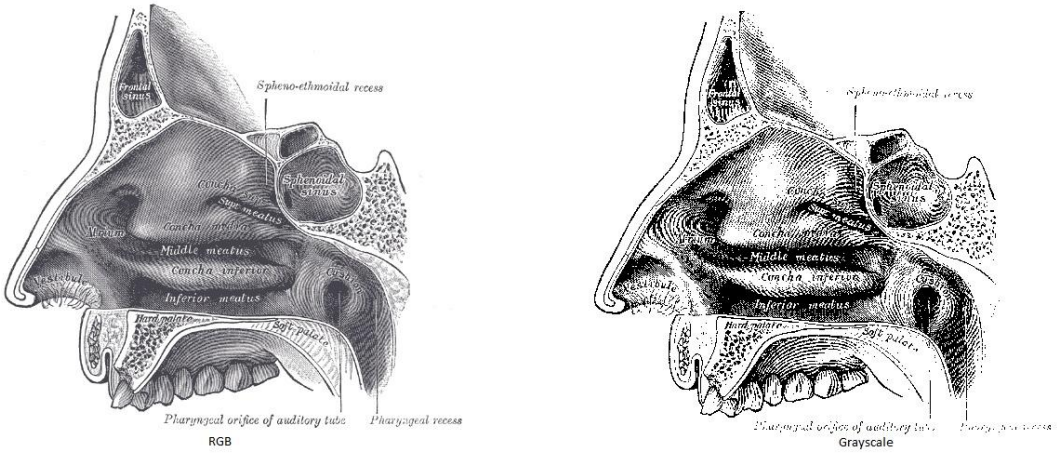
Gambar 3.2 Proses *resize* gambar

Algoritma *resize*:

```
fun
O2D_ContainerResizingEvent(Window,Container,Width,Height)=
  if Width && Height
  then
  {
    /* Reallocate the working bitmap */
      let [Container.CoWorkingBitmap
                    (_CRbitmap Container.CoChannel Width
Height)]
      -> [OldWorkingBitmap NewWorkingBitmap] in
      (
            _CPbitmap16 NewWorkingBitmap 0 0 OldWorkingBitmap
0 0
Container.CoXSize Container.CoYSize nil;
            set Container.CoWorkingBitmap=NewWorkingBitmap;
            set Container.CoXSize=Width;
            set Container.CoYSize=Height;/*proses resize
gambar*/
            _DSbitmap OldWorkingBitmap
      );
      /* Repaint the window */
    O2D_RedrawContainer Container 1;
      /* Call the user window resizing reflex function */
      exec Container.CoContainerResizingReflex with
[Container Width Height]
  }
  else
    nil
```

4. Grayscale

Grayscale merupakan proses mengubah gambar menjadi monocrom dengan nilai intensitas 0 untuk hitam, 255 untuk putih, dan abu-abu untuk nilai yang berada panda rentang 0 sampai 255. Dimana rumus secara garis besarnya adalah *new pixel = (Red + Green + Blue)*/3. Hasilnya terlihat seperti gambar 3.3.



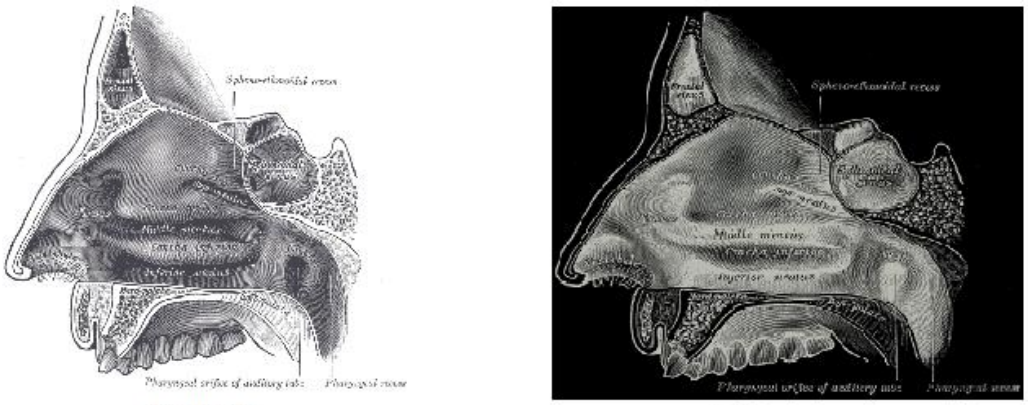
Gambar 3.3 Proses grayscale gambar

Algoritma *Grayscale*:

```
fun BMP_ChangeGrayscale(rgb,grayscale)=
{
  if grayscale==nil then
  {
    set rgb.RGBgrayscale=nil;
    set rgb.RGBObj2D.O2Dw=0;
    set rgb.RGBObj2D.O2Dh=255;
  }
  else
    let _GETgrayscaleSize grayscale -> [r g b] in
    {
      set rgb.RGBgrayscale=grayscale;
      set rgb.RGBObj2D.O2Dw=r;
      set rgb.RGBObj2D.O2Dh=g;
      set rgb.RGBObj2D.O2Dh=b;/*proses grayscale gambar*/
    };
```

5. Histogram

Proses *histogram* adalah suatu perataan warna abu-abu, dimana distribusi derajat keabuan pada suatu gambar dibuat rata. Seperti terlihat pada gambar 3.4.



Gambar 3.4 Proses histogram gambar

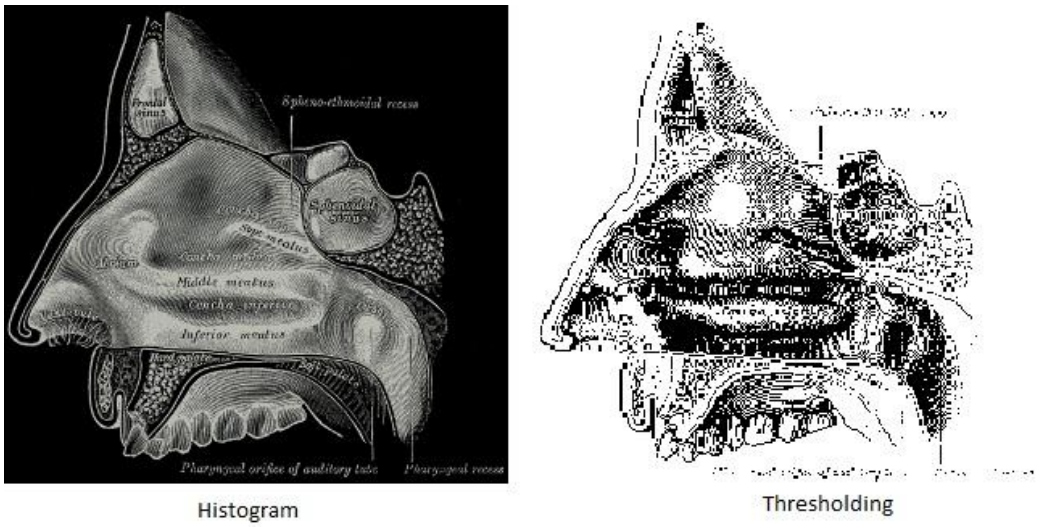
Algoritma *Histogram*:

```
fun O2D_ClearAreaIncludingObject(bmp,obj,color)=
  if color!=nil then
  {
       let O2D_RectangleIncludingObject obj -> Area in
       let SizeRectangle Area -> [AreaXSize AreaYSize] in
       _DRAWrectangle bmp
                      Area.RctHG.iptX
                      Area.RctHG.iptY
                      AreaXSize+1 AreaYSize+1
                      DRAW_INVISIBLE 0 0
                      DRAW_SOLID color
  }
  else nil
 (A,int N,int M)
       for(i=0;i<=255;i++)
             Hist[i]=0
       endfor
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/*proses histogram gambar*/
             endif
       endfor
endfor
endfor
O2D_DisplayObjectsInArea      Container      Node.DNNextBrother
AreaToRedraw;
```

6. Ambang Batas (Threshold)

Untuk *Aruco Library, threshold* ditentukan besarnya agar gambar yang diambil dapat masuk kedalam *pool* untuk diproses lebih lanjut.

Proses *threshold* ini yang menjadi acuan untuk proses selanjutnya yaitu pencocokkan pola gambar. Dalam *Aruco Library* ukuran *threshold* dapat diatur sesuai keakuratan gambar pada saat proses pencocokkan pola dengan jarak tertentu. Seperti terlihat pada gambar 3.5.



Gambar 3.5 Proses threshold gambar

Gambar *threshold* tersebut kemudian diproses untuk mendeteksi gambar dan pencocokkan pola menggunakan metode *Euclidean Distance* yang telah disediakan pada *OpenSpace.*

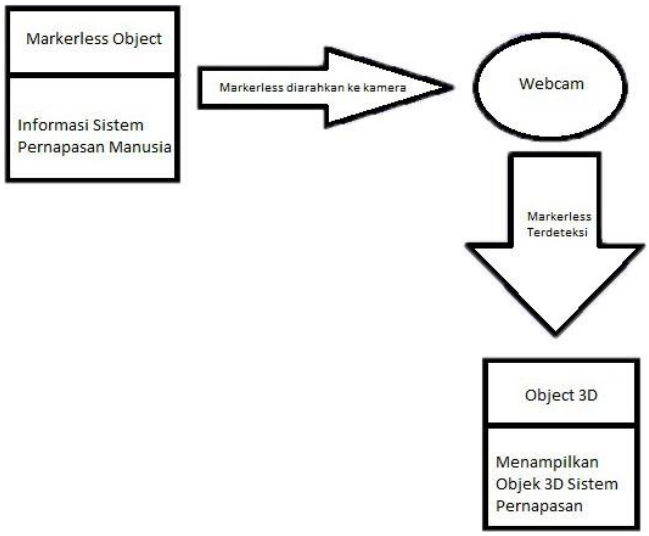
Algoritma *threshold:*

```
{ /* style pointille ou arrondi */
        let nil -> ListPoints in
        let (if StyleTrait==STYLE_POINTILLE_1 then (5::3::nil)
             else   if   StyleTrait==STYLE_POINTILLE_2   then
(2::2::nil)
             else   if   StyleTrait==STYLE_POINTILLE_3   then
(7::2::2::2::nil)
```

```
            else    if    StyleTrait==STYLE_POINTILLE_4    then
(5::2::2::2::2::2::nil)
            else nil
           ) -> listaffichage in
        let [1 0 0] -> [Plein i indice] in
        let   (if   CoinArrondi   then   (if   Width>Height   then
[Width/4 Height/4] else [Height/4 Width/4]) else [0 0]) -> [a
b] in
        let EvalPointsLigne (MkInt2DPoint [a 0]) (MkInt2DPoint
[Width-a-1 0]) -> hs in /* ligne horizontale haute*/
        let   EvalPointsLigne   (MkInt2DPoint   [Width-1   b])
(MkInt2DPoint [Width-1 Height-b-1]) -> bd in /*ligne verticale
droite*/
        let   EvalPointsLigne   (MkInt2DPoint   [a   Height-1])
(MkInt2DPoint   [Width-a-1   Height-1])   ->   hb   in   /*ligne
horizontale basse *//*proses threshold gambar*/
        let EvalPointsLigne (MkInt2DPoint [0 b]) (MkInt2DPoint
[0 Height-b-1]) -> bg in /* ligne verticale gauche*/
        if CoinArrondi then
        {
          let EvalPointsEllipse  (MkInt2DPoint [a b])
                                 (MkInt2DPoint [Width-a b])
                                 (MkInt2DPoint [a Height-b])
                                 (MkInt2DPoint        [Width-a
Height-b])
                                a b
          -> [supg infg supd infd] in
          PutAllGroupPixelAlternative  BmpRes  TraitColor  1  0
listaffichage 0 (hs::supd::bd::infd::hb::infg::bg::supg::nil)
        }
        else PutAllGroupPixelAlternative BmpRes TraitColor 1 0
listaffichage 0 (hs::bd::hb::bg::nil);
        BmpRes
  };
```

3.1.2.2 Tracking Markerless

Proses *tracking* yang terjadi adalah dengan pembacaan ID gambar pada *database* dengan tampilan pada *marker*. Alur pembacaan *markerless* digambarkan seperti pada gambar 3.6.



Gambar 3.6 Alur *Tracking Markerless*

Berikut ini adalah algoritma untuk menampilkan setiap objek pada aplikasi sistem pernafasan manusia.

– **Algoritma Menampilkan Object**

Algoritma untuk menampilkan *object* 3D hidung adalah sebagai berikut :

```
<instance py="11" px="56" name="AR+marker+hidung"
comment="">
<param name="object"><![CDATA[9.hidung]]></param>/*proses
menampilkan object 3D hidung*/
<param name="msize"><![CDATA[0.080000]]></param>
<param name="id"><![CDATA[0.000000]]></param>
<param
name="path"><![CDATA[hidung_15499_lg_marker.jpg]]></param>
<param name="maxfeatures"><![CDATA[200.000000]]></param>
<param name="enable"><![CDATA[1]]></param>
<param name="track"><![CDATA[1]]></param>
</instance>
```

– **Algoritma Menampilkan Object**

Algoritma untuk menampilkan *object* 3D laring adalah sebagai berikut :

```
<instance py="11" px="56" name="AR+marker+laring" comment="">
<param name="object"><![CDATA[9.laring]]></param>/*proses
menampilkan object 3D laring*/
<param name="msize"><![CDATA[0.
<param name="msize"><![CDATA[0.080000]]></param>
<param name="id"><![CDATA[0.000000]]></param>
<param
name="path"><![CDATA[larinx_15499_lg_marker.jpg]]></param>
<param name="maxfeatures"><![CDATA[200.000000]]></param>
<param name="enable"><![CDATA[1]]></param>
<param name="track"><![CDATA[1]]></param>
</instance>
```

– **Algoritma Menampilkan Object**

Algoritma untuk menampilkan *object* 3D trakea adalah sebagai berikut :

```
<instance py="11" px="56" name="AR+marker+trakea" comment="">
<param name="object"><![CDATA[9.trakea]]></param>/*proses
menampilkan object 3D trakea*/
<param name="msize"><![CDATA[0.
<param name="msize"><![CDATA[0.080000]]></param>
<param name="id"><![CDATA[0.000000]]></param>
<param
name="path"><![CDATA[trachea_15499_lg_marker.jpg]]></param>
<param name="maxfeatures"><![CDATA[200.000000]]></param>
<param name="enable"><![CDATA[1]]></param>
<param name="track"><![CDATA[1]]></param>
</instance>
```

– **Algoritma Menampilkan Object**

Algoritma untuk menampilkan *object* 3D bronkus adalah sebagai berikut :

```
<instance py="11" px="56" name="AR+marker+bronkus" comment="">
<param name="object"><![CDATA[9.bronkus]]></param>/*proses
menampilkan object 3D bronkus*/
<param name="msize"><![CDATA[0.
<param name="msize"><![CDATA[0.080000]]></param>
<param name="id"><![CDATA[0.000000]]></param>
<param
name="path"><![CDATA[bronchus_15499_lg_marker.jpg]]></param>
<param name="maxfeatures"><![CDATA[200.000000]]></param>
<param name="enable"><![CDATA[1]]></param>
<param name="track"><![CDATA[1]]></param>
</instance>
```

– **Algoritma Menampilkan Object**

Algoritma untuk menampilkan *object* 3D paru-paru kanan adalah sebagai berikut :

```
<instance py="10" px="32" name="AR+marker+parukanan"
```

```
comment="">
<param name="object"><![CDATA[11.paru kanan]]></param>/*proses
menampilkan object 3D paru-paru kanan*/
<param name="msize"><![CDATA[0.
<param name="msize"><![CDATA[0.080000]]></param>
<param name="id"><![CDATA[0.000000]]></param>
<param name="path"><![CDATA[paru-kanan_marker.jpg]]></param>
<param name="maxfeatures"><![CDATA[400.000000]]></param>
<param name="enable"><![CDATA[1]]></param>
<param name="track"><![CDATA[1]]></param>
</instance>
```

– **Algoritma Menampilkan Object**

Algoritma untuk menampilkan *object* 3D paru-paru kiri adalah sebagai berikut:

```
<instance     py="10"     px="45"     name="AR+marker+parukiri"
comment="">
<param name="object"><![CDATA[10.paru kiri]]></param>/*proses
menampilkan object 3D paru-paru kiri*/
<param name="msize"><![CDATA[0.
<param name="msize"><![CDATA[0.080000]]></param>
<param name="id"><![CDATA[0.000000]]></param>
<param name="path"><![CDATA[paru-kiri_marker.jpg]]></param>
<param name="maxfeatures"><![CDATA[400.000000]]></param>
<param name="enable"><![CDATA[1]]></param>
<param name="track"><![CDATA[1]]></param>
</instance>
```

– **Algoritma Menampilkan Video**

Algoritma untuk menampilkan *video* adalah sebagai berikut:

```
<plugin name="Video" source="media%2fvideo%2fvideo%2exml">
                <instance py="33" px="92" name="Video+respirasi"
comment="">
                    <param
name="istexture"><![CDATA[0]]></param>
                    <param name="object"/>
                    <param name="material"/>
                    <param           name="technique"><![CDATA[-
1]]></param>
                    <param name="pass"><![CDATA[-1]]></param>
                    <param name="texture"><![CDATA[-1]]></param>
                    <param name="isurl"><![CDATA[0]]></param>
                    <param
name="url"><![CDATA[http://]]></param>
<paramname="path"><![CDATA[mekanismenapas.flv.mp4]]></param>/*pr
oses menampilkan video*/
<param name="msize"><![CDATA[0.
                    <param name="autoplay"><![CDATA[1]]></param>
                    <param name="loop"><![CDATA[0]]></param>
                    <param name="controls"><![CDATA[1]]></param>
```

```
                    <param name="volume"><![CDATA[50]]></param>
                    <param
name="background"><![CDATA[0]]></param>
                    <param name="posx"><![CDATA[0]]></param>
                    <param name="posy"><![CDATA[0]]></param>
                    <param name="width"><![CDATA[320]]></param>
                    <param name="height"><![CDATA[200]]></param>
                    <param name="xpercent"><![CDATA[0]]></param>
                    <param name="ypercent"><![CDATA[0]]></param>
                    <param name="wpercent"><![CDATA[0]]></param>
                    <param name="hpercent"><![CDATA[0]]></param>
                    <param name="xoffset"><![CDATA[0]]></param>
                    <param name="yoffset"><![CDATA[0]]></param>
                    <param name="woffset"><![CDATA[0]]></param>
                    <param name="hoffset"><![CDATA[0]]></param>
                    <param name="xalign"><![CDATA[0]]></param>
                    <param name="yalign"><![CDATA[0]]></param>
                    <param
                    <param
name="texturequality"><![CDATA[100]]></param>
                    <param name="zorder"><![CDATA[0]]></param>
                    <param
name="transparency"><![CDATA[0]]></param>
                    <param name="focus"><![CDATA[0]]></param>
                    <param
name="toponfocus"><![CDATA[0]]></param>
                    <param
name="allwaysontop"><![CDATA[0]]></param>
                    <param
name="enablekeyboard"><![CDATA[1]]></param>
                    <param
name="enablemouse"><![CDATA[1]]></param>
                    <param name="init"><![CDATA[0]]></param>
                </instance>
            </plugin>
```

– **Algoritma Menampilkan Text**

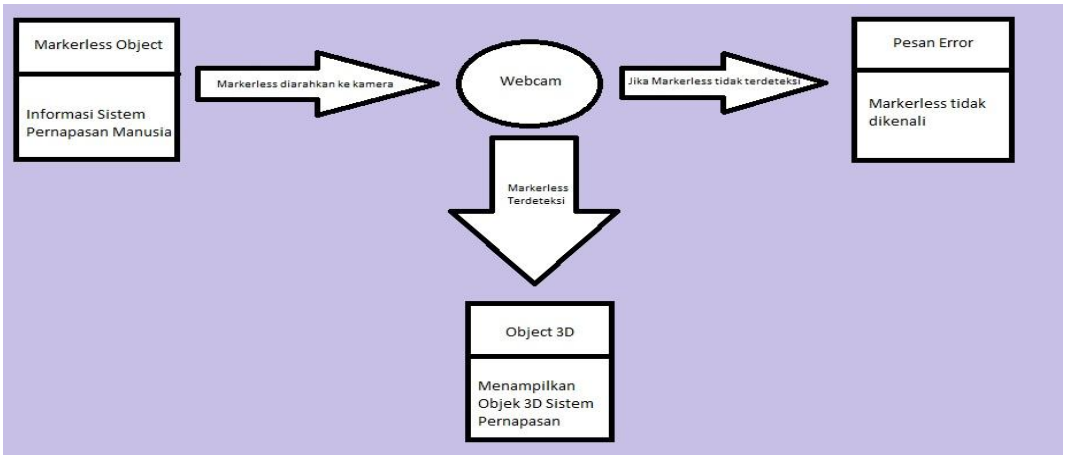
Algoritma untuk menampilkan *text* pada obyek 3D adalah sebagai berikut:

```
<plugin name="object+click"
source="objects%2fobjectclick%2fobjectclick%2exml">/*proses
menampilkan object text*/
<param name="msize"><![CDATA[0.
                <instance            py="19"            px="7"
name="object+click+parukanan" comment="">
                    <param
name="object"><![CDATA[6.mesh005]]></param>
                    <param
name="material"><![CDATA[mesh002]]></param>
                    <param
name="enablemat"><![CDATA[1]]></param>
                    <param name="cursor"><![CDATA[1]]></param>
                    <param name="bubble"><![CDATA[…]]></param>
                    <param name="ignore"><![CDATA[0]]></param>
                    <param name="format"><![CDATA[0]]></param>
```

```
                    <param name="enable"><![CDATA[1]]></param>
                </instance>
```

3.1.3 Analisis Arsitektur Aplikasi

Webcam pada pembuatan aplikasi ini berperan penting bagi perangkat lunak untuk menangkap *markerless object.* Komputer akan mendeteksi pola *markerless* yang sesuai kemudian *markerless* dapat ditentukan. Dengan informasi yang didapat, obyek virtual digabungkan dengan *markerless* yang didapat dari *webcam* dan merendernya sesuai dengan informasi posisi yang diperoleh dari *markerless object* tersebut. Proses tersebut terjadi secara *real-time* sehingga model virtual yang tampil di media *display* akan mengikuti pergerakan *markerless object*. Berikut merupakan gambaran arsitektur aplikasi yang akan dibuat dapat dilihat pada gambar 3.7.



Gambar 3.7 Arsitektur Aplikasi Sistem pernafasan Manusia

3.1.4 Analisis Metode

Terdapat beberapa metode yang biasa digunakan pada *augmented reality*, salah satunya adalah *Markerless Augmented Reality*, dimana metode ini tanpa menggunakan frame marker sebagai obyek yang dideteksi. Dengan adanya *Markerless Augmented Reality*, maka penggunaan *marker* sebagai *tracking object* yang selama ini menghabiskan ruang, akan digatikan dengan gambar, atau permukaan apapun yang berisi dengan tulisan, logo atau gambar sebagai *tracking object* agar dapat langsung melibatkan obyek yang dilacak tersebut sehingga dapat

terlihat hidup dan interaktif, juga tidak lagi mengurangi efisiensi ruang dengan adanya *marker.*

Pelacakan *markerless*, posisi kamera dan orientasi kamera dihitung antara kamera dan dunia nyata tanpa referensi apapun, hanya menggunakan titik-titik fitur alami (edge, corner, garis atau model 3D). Metode *markerless* memerlukan model atau gambar referensi untuk inisialisasi. Berikut adalah tahap pembacaan *markerless* pada metode *Markerless Augmented Reality:*

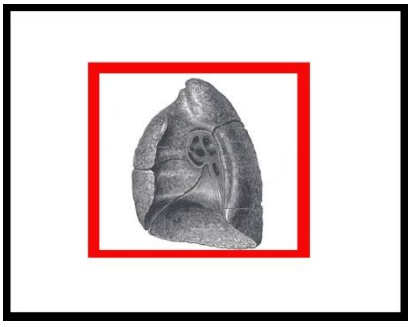
1. Tahap Pembacaan Markerless

– Menerapkan ambang batas (*Tresholding*) *adaptif* untuk memperoleh batas. Seperti terlihat pada gambar 3.8.



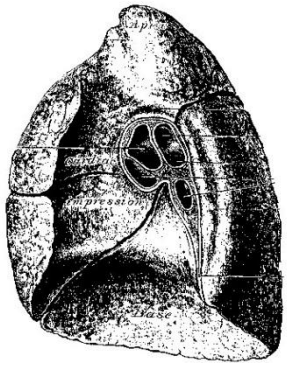
Gambar 3.8 Menerapkan Batas (*Tresholding*)

– Cari kontur. Setelah itu, tidak hanya marker terdeteksi, tetapi juga banyak batas yang tidak diinginkan. Sisa dari proses ini bertujuan untuk menyaring perbatasan yang tidak diinginkan. Seperti terlihat pada gambar 3.9.



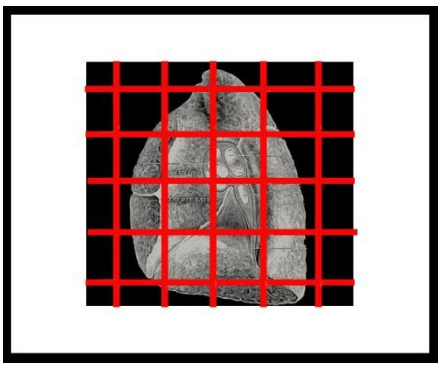
Gambar 3.9 Mencari Kontur Markerless

– Batas daerah marker menggunakan *Otsu Tresholding*, yang mengasumsikan distribusi bimodal dan menemukan ambang batas yang menghasilkan varian ekstra kelas. Seperti terlihat pada gambar 3.10.



Gambar 3.10 Otsu Tresholding

– Identifikasi kode internal. *Aruco Library* akan memasukkan kode-kode dalam bentuk penanda kedalam marker yang telah diproses. Penanda ini dibagi kedalam kotak 6x6, dimana sel-sel 5x5 berisi informasi ID. Harus dipastikan garis hitam ada pada marker. Setelah itu, *Aruco Library* akan membaca sel 5x5 dan memeriksa apakah sel-sel itu menyediakan kode yang *valid*. Seperti terlihat pada gambar 3.11.

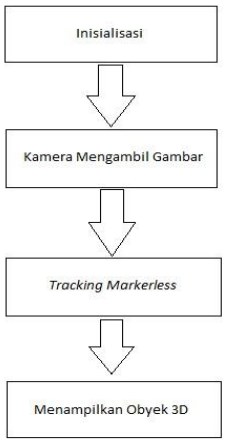


Gambar 3.11 Pembacaan sel pada Markerless

– Marker yang telah *valid*, setiap sudut marker diperbaiki dengan cara *interpolasi*. Parameter kamera yang telah tersedia akan dihitung dengan *extrincsics* marker ke kamera.

3.1.5 Analisis Perancangan Aplikasi

Analisis perancangan aplikasi memerlukan beberapa tahapan yang harus dijalankan oleh aplikasi, tahapan tersebut seperti digambarkan pada gambar 3.12 berikut:



Gambar 3.12 Diagram alir system augmented reality

Keterangan:

a. Inisialisasi

   Pada tahap ini aplikasi akan mendeteksi ketersediaan kamera pada perangkat keras user.

b. Kamera mengambil gambar

   Pada tahapan ini, kamera akan menangkap langsung gambar dari dunia nyata.

c. Tracking markerless

   Pelacakan *markerless*, posisi kamera dan orientasi kamera dihitung antara kamera dan dunia nyata tanpa referensi apapun, hanya menggunakan titik-titik fitur alami (edge, corner, garis atau model 3D). Metode *markerless* memerlukan langkah priori manual, serta model atau gambar referensi untuk inisialisasi.

d. Menampilkan obyek 3D

   Dengan menggunakan kamera dan *markerless*, obyek virtual dapat muncul diatas obyek *markerless* seperti berada di dunia nyata.

3.1.6 Analisis Materi Dalam Aplikasi AR Sistem pernafasan Manusia

Materi-materi sistem pernafasan manusia yang akan ditampilkan dalam aplikasi ini diantaranya meliputi:

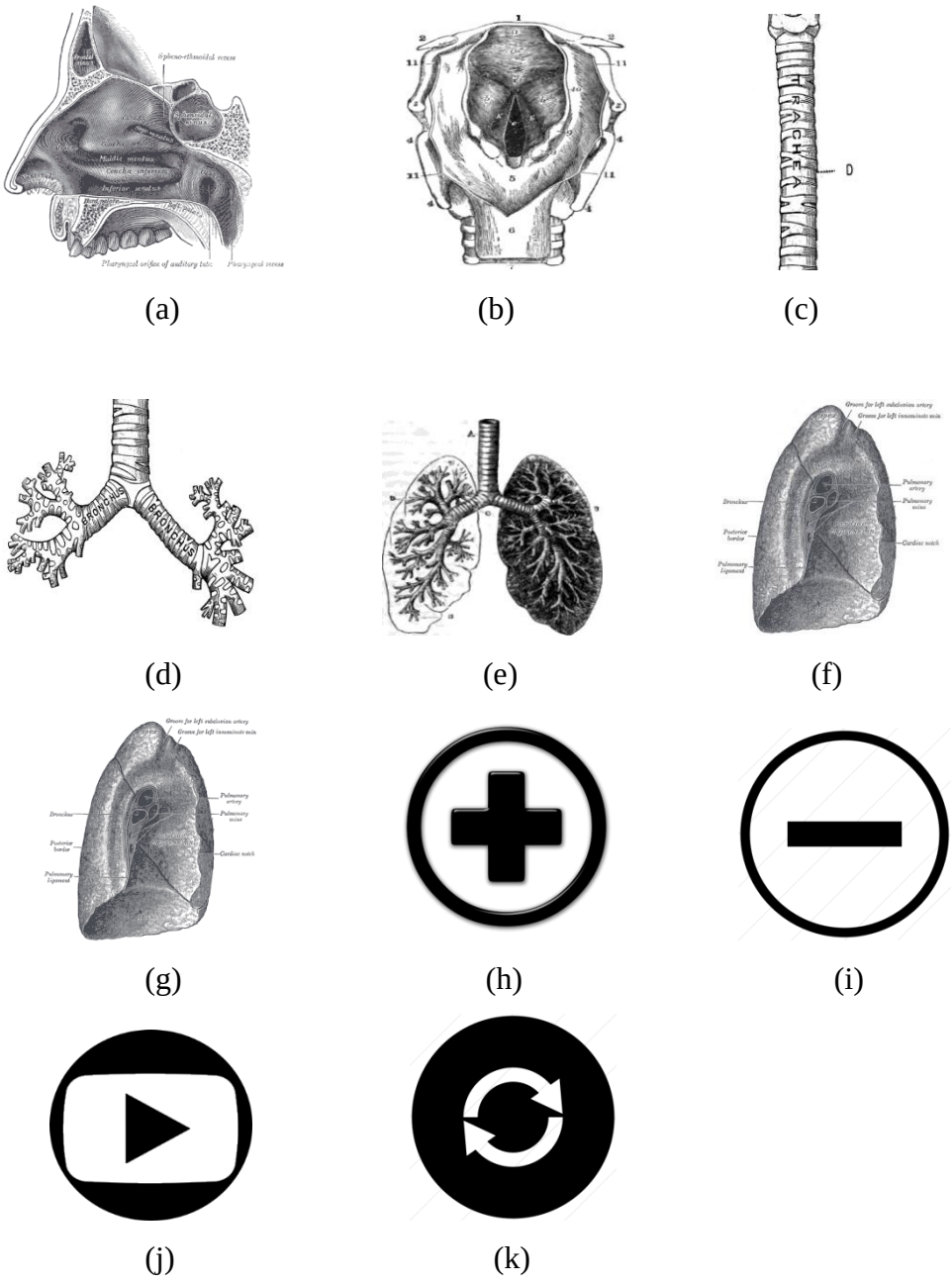
1. Udara
   a. Fungsi udara
   b. Elemen udara pada pernafasan
      – Oksigen
      – Karbon dioksida
2. Alat Pernafasan Manusia
   a. Hidung
   b. Faring
   c. Laring
   d. Batang Tenggorok (Trakea)
   e. Cabang Batang Tenggorok (Bronkia)
   f. Paru-Paru
3. Animasi Sistem pernafasan Manusia

3.1.7 Perancangan Markerless

Metode *markerless* merupakan pengembangan dari metode *marker* pada *Augmented Reality*, dimana *Markerless Augmented Reality* tanpa menggunakan frame marker sebagai obyek yang dideteksi.

Dengan adanya *Markerless Augmented Reality*, maka, penggunaan *marker* sebagai *tracking object* yang selama ini menghabiskan ruang, akan digantikan dengan gambar, atau permukaan apapun yang berisi dengan tulisan, logo sebagai *tracking object* agar dapat langsung melibatkan obyek yang dilacak tersebut, sehingga dapat terlihat hidup dan interaktif, juga tidak lagi mengurangi efisiensi ruang dengan adanya *marker*.

Tracking object markerless yang digunakan di dalam aplikasi AR sistem pernafasan pada manusia seperti pada gambar 3.13 berikut:



Gambar 3.13 *Tracking Object Markerless* aplikasi sistem pernafasan manusia

Keterangan gambar :

Gambar a : *Tracking Object* untuk animasi 3D anatomi hidung manusia

Gambar b : *Tracking Object* untuk animasi 3D laring manusia

Gambar c : *Tracking Object* untuk animasi 3D trakea manusia

Gambar d : *Tracking Object* untuk animasi 3D bronkus manusia
Gambar e : *Tracking Object* untuk animasi 3D sistem pernafasan manusia
Gambar f : *Tracking Object* untuk animasi 3D paru-paru kiri
Gambar g : *Tracking Object* untuk animasi 3D paru-paru kanan
Gambar h : *Tracking Object* untuk kontrol obyek 3D *zoom-in*
Gambar i : *Tracking Object* untuk kontrol obyek 3D *zoom-out*
Gambar j : *Tracking Object* untuk *play* video
Gambar k : *Tracking Object* untuk kontrol obyek 3D rotasi

3.1.7.1 Real-Time Markerless Tracking Augmented Reality

Teknik ini memungkinkan obyek 3D mengikuti pergerakan *tracking object* yang telah disediakan, biasanya teknik ini digunakan untuk fungsi *zoom-in zoom-out* dan rotasi obyek 3D dengan menggunakan metode *markerless* [6].

Metode ini dimaksudkan untuk mengganti fungsi *zoom-in* dan *zoom-out* serta fungsi rotasi menggunkan *marker*, sehingga tanpa ada *marker* khusus, obyek 3D bisa diputar dan dilakukan fungsi pembesaran dan pengecilan obyek. Tahap ini, akan dicari model *indicator* dengan pengaturan *parameter* berupa:

a. Permukaan 2D
b. Koordinat *x, y, z*
c. Jarak antar *markerless*

3.1.8 Analisis Kebutuhan Non Fungsional

Analisis kebutuhan non fungsional merupakan analisis yang dibutuhkan untuk menentukan spesifikasi kebutuhan sistem. Spesifikasi ini juga meliputi elemen atau komponen-komponen apa saja yang dibutuhkan untuk sistem yang akan dibangun sampai dengan sistem tersebut diimplementasikan. Analisis kebutuhan ini juga menentukan spesifikasi masukan yang diperlukan sistem, keluaran yang akan dihasilkan sistem, dan proses yang dibutuhkan untuk mengolah masukan sehingga menghasilkan suatu keluaran yang diinginkan.

Pada analisis kebutuhan non fungsional ini dijelaskan analisis kebutuhan perangkat lunak, analisis kebutuhan perangkat keras, dan analisis pengguna.

3.1.8.1 Analisis Kebutuhan Perangkat Lunak

Perangkat lunak yang dibutuhkan untuk membangun aplikasi multimedia presentasi pembelajaran berbasis *augmented reality* untuk sistem pernafasan manusia ini adalah sebagai berikut:

1) *Microsoft windows* 7 Ultimate
2) *Autodesk 3D Studio Max 2010*
3) Ogre3D untuk *export* file 3D dari Autodesk 3D *Studio Max*
4) Scol_plugin untuk OpenSpace3D
5) OpenSpace3D Editor
6) *Adobe Photoshop 7*
7) Adobe Flash CS 3
8) StarUML

3.1.8.2 Analisis Kebutuhan Perangkat Keras

Kebanyakan untuk aplikasi yang berbasis *augmented reality* pasti membutuhkan grafis video yang relatif tinggi. Hal ini dikarenakan model-model grafis yang digunakan menggunakan teknik *rendering*.

Berdasarkan studi literatur terhadap proyek-proyek yang mengembangkan teknologi *augmented reality*, maka diperoleh spesifikasi minimum perangkat keras untuk pengembang teknologi *augmented reality* seperti berikut ini:

1) Processor Intel® Core™ i3-2367M
2) *Random Access Memory* (RAM) 4 GB
3) *Harddisk* dengan space 167 MB
4) VGA Intel® HD Graphics 3000
5) VGA NVidia Geforce GT 740M 1GB
6) *Webcam* 30 fps 1,3 Mega Pixel

3.1.8.3 Analisis Kebutuhan Pengguna

Analisis pengguna dimaksudkan untuk mengetahui siapa saja pengguna aplikasi. Pengguna aplikasi ini sendiri yaitu guru yang mengerti dan memahami cara kerja aplikasi ini, sehingga dapat menggunakan aplikasi

yang akan dibangun. Selain itu, aplikasi ini juga bisa digunakan langsung oleh siswa sendiri agar para siswa bisa lebih berinteraksi langsung dengan apa yang diajarkan oleh guru.

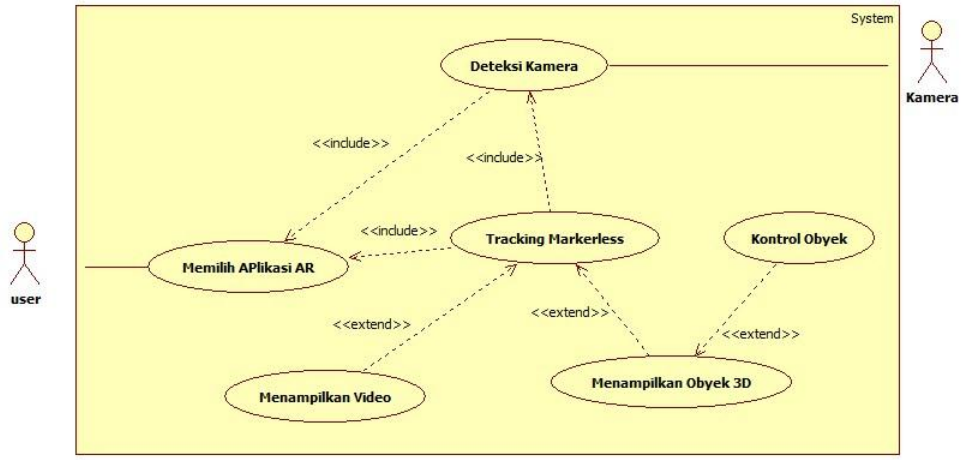
3.1.9 Analisis Kebutuhan Fungsional

Analisis kebutuhan fungsional ini dimodelkan dengan menggunakan UML (*Unified Modeling Language*). Dimana tahap-tahap perancangan yang dilakukan dalam membangun aplikasi multimedia presentasi pembelajaran berbasis *augmented reality* untuk sistem pernafasan antara lain *Use Case Diagram, Class Diagram, Sequence Diagram*, dan *Activity Diagram*.

3.1.9.1 Use Case Diagram

Diagram Use Case merupakan pemodelan untuk kelakuan (*behaviour*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat [7].

Berikut ini adalah perancangan proses-proses yang terdapat pada aplikasi multimedia presentasi pembelajaran berbasis *augmented reality* untuk sistem pernafasan, yang digambarkan dengan Use Case Diagram yang dapat dilihat pada gambar 3.14 berikut.



Gambar 3.14 *Use Case Diagram*

1. Definisi *Actor*

Actor yaitu elemen pada *use case* yang mempresentasikan seseorang atau sesuatu yang berinteraksi dengan sistem. Definisi *Actor* diterangkan pada Tabel 3.1.

Tabel 3.1 Definisi *Actor*

No	Actor	Deskripsi
1	User	Orang yang menggunakan aplikasi
2	Kamera	Alat input yang digunakan aplikasi

2. Definisi *Use Case*

Definisi *Use Case* adalah gambaran fungsionalitas dari suatu sistem, sehingga *user* sistem paham mengenai kegunaan sistem yang akan dibangun. Definisi *Use Case* diterangkan pada Tabel 3.2.

Tabel 3.2 Definisi *Use Case*

No	Use Case	Deskripsi
1	Aplikasi AR	Proses untuk memulai aplikasi AR
2	Menampilkan Animasi Obyek 3D	Proses dimana animasi obyek-obyek 3D yang ada dalam aplikasi akan muncul
3	Menampilkan Video	Proses dimana video-video yang ada dalam aplikasi akan muncul
4	Deteksi Kamera	Proses dimana aplikasi akan mendeteksi ketersediaan kamera
5	*Tracking* Markerless	Proses dimana kamera akan mendeteksi obyek markerless
6	Kontrol Obyek	Proses untuk memperbesar/perkecil serta putar obyek

3. Skenario *Use Case*

Skenario *Use Case* menggambarkan alur penggunaan sistem dimana setiap skenario digambarkan dari sudut pandang aktor, seseorang, atau piranti yang berinteraksi dengan perangkat lunak dalam berbagai cara.

Tabel 3.3 Skenario Aplikasi AR

Nama *Use Case*	Aplikasi AR
Nomor	1

Aktor	User, kamera	
Kondisi Awal	User berada ditampilan menu aplikasi	
Reaksi Aktor		**Reaksi Sistem**
1. *Markerless object* diarahkan ke kamera		2. Sistem membuka aplikasi AR sistem pernafasan. 3. Sistem mendeteksi ketersediaan kamera. 4. Sistem siap mendeteksi *markerless object*
Kondisi Akhir	Aplikasi menampilkan jendela deteksi *marker*	

Tabel 3.4 Skenario Menampilkan Video

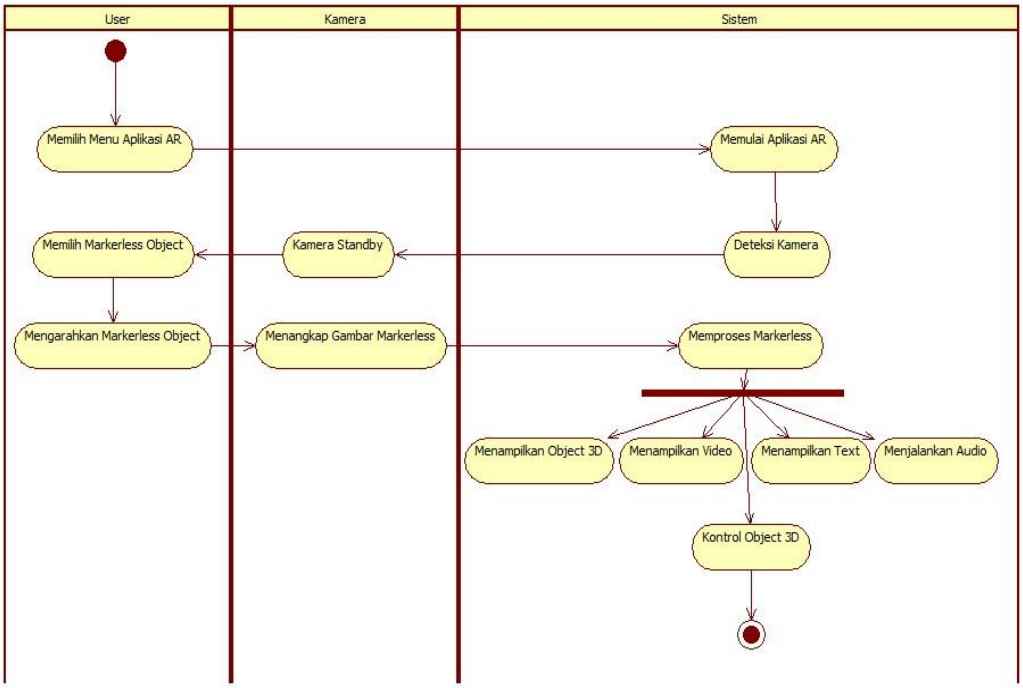
Nama *Use Case*	Menampilkan Video	
Nomor	5	
Aktor	User, kamera	
Kondisi Awal	Kamera siap mendeteksi marker	
Reaksi Aktor		**Reaksi Sistem**
1. User menunjukan *markerless object* ke kamera.		2. Kamera menangkap pattern gambar dari setiap *markerless.* 3. Sistem menampilkan video-video tentang sistem pernafasan.
Kondisi Akhir	Video-video yang ada dalam aplikasi muncul	
Pengecualian	1. *Pattern markerless object* tidak terdeteksi	

Tabel 3.5 Skenario Kontrol Obyek

Nama *Use Case*	Kontrol Obyek	
Nomor	8	
Aktor	User, kamera	
Kondisi Awal	Obyek 3D tampil dilayar	
Reaksi Aktor		**Reaksi Sistem**
1. User menunjukan *markerless object* kontrol ke kamera.		2. Kamera menangkap pattern dari *markerless* kontrol obyek. 3. Sistem mengubah ukuran dan rotasi dari obyek-obyek 3D yang muncul.
Kondisi Akhir	Obyek-obyek 3D yang muncul dapat diperbesar-diperkecil dan diputar.	
Pengecualian	1. *Pattern markerless control object* tidak terbaca	

3.1.9.2 Activity Diagram

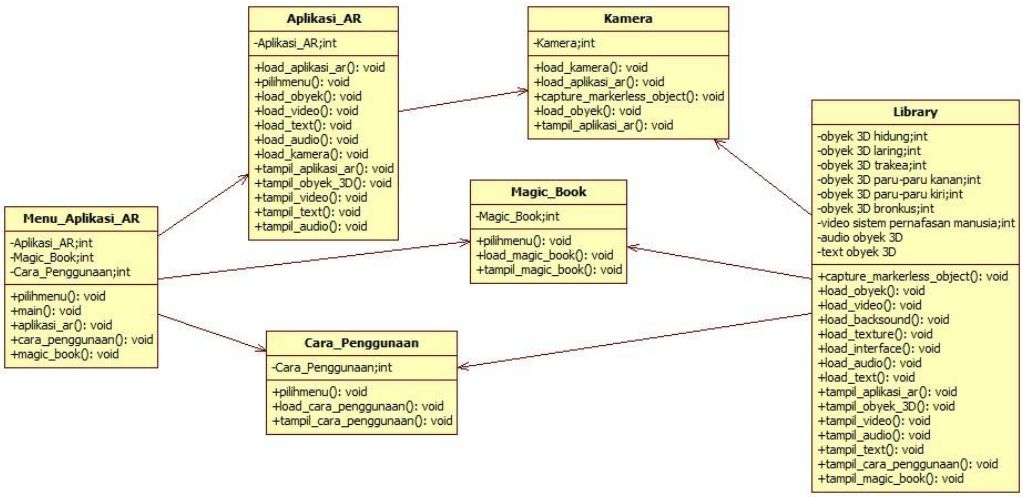
Diagram aktivitas atau *Activity Diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis [7]. Penggambaran *activity* diagram memiliki kemiripan dengan *flowchart diagram. Activity diagram* memodelkan *event-event* yang terjadi pada *UseCase* dan digunakan untuk pemodelan aspek dinamis dari sistem. Seperti terlihat pada gambar 3.15 berikut.



Gambar 3.15 *Activity Diagram* Aplikasi AR

3.1.9.3 Class Diagram

Diagram kelas atau *class diagram* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang dibuat untuk membangun sistem [7].Berikut ini adalah perancangan struktur sistem yang terdapat pada aplikasi AR sistem pernafasan manusia yang digambarkan dengan *Class Diagram* yang dapat dilihat pada gambar 3.16 berikut.



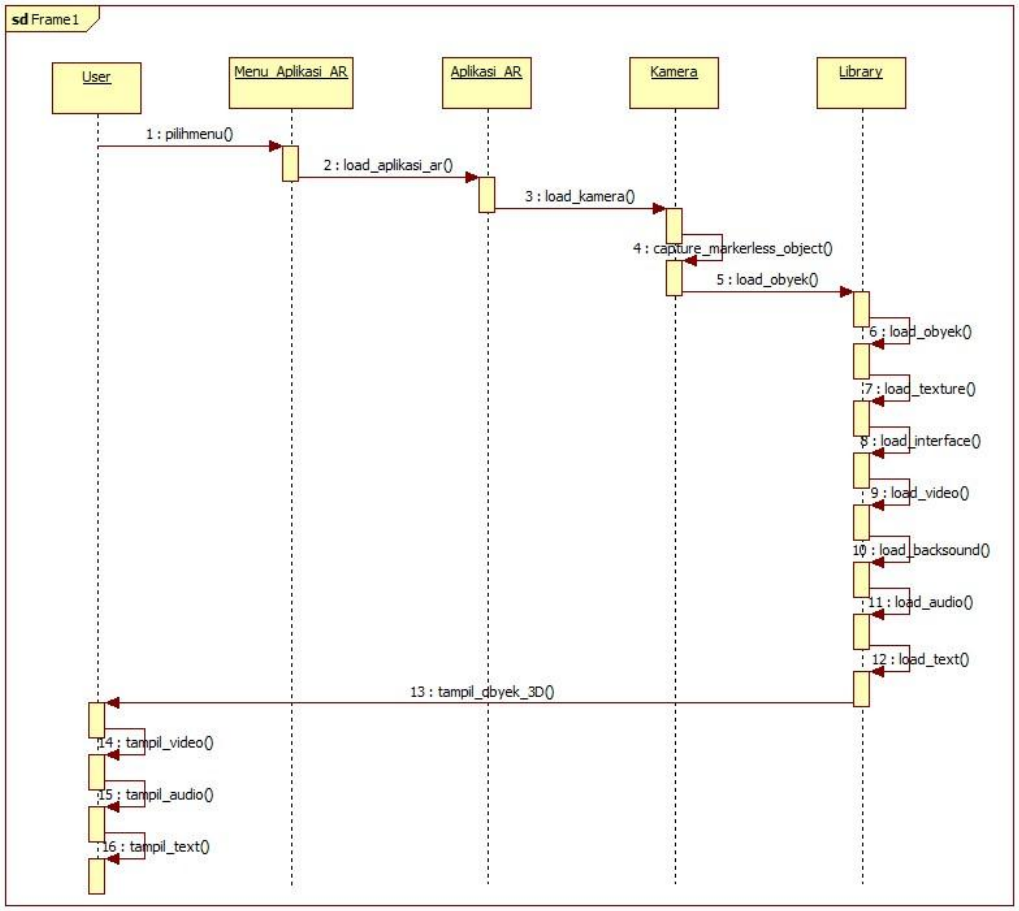
Gambar 3.16 *Class Diagram* Aplikasi AR Sistem Pernafasan Manusia

Tabel 3.6 Deskripsi Class Diagram

Class	Jenis Class	Deskripsi
Menu_Aplikasi_AR	Interface	Class yang berfungsi menampilkan halaman untuk memilih menu pada aplikasi
Cara_Penggunaan	Interface	Class yang berfungsi menampilkan halaman untuk cara menggunakan aplikasi
Magic_Book	Interface	Class yang berfungsi menampilkan halaman yang menjelaskan media pembelajaran baru berbentuk magic book
Aplikasi_AR	Control	Class yang mempunyai fungsionalitas untuk menjalankan dan menampilkan *output Augmented Reality*
Kamera	Control	Class yang mempunyai fungsionalitas untuk menjalankan kamera untuk deteksi gambar
Library	Entity	Class yang berisikan data-data obyek yang ada pada aplikasi

3.1.9.4 Sequence Diagram

Diagram *sequence* atau *Sequence Diagram* menggambarkan kelakuan obyek pada *use case* dengan mendeskripsikan waktu hidup obyek dan *message* yang dikirimkan dan diterima antar obyek [7]. Seperti terlihat pada gambar 3.17 berikut.



Gambar 3.17 *Sequence Diagram* Aplikasi AR

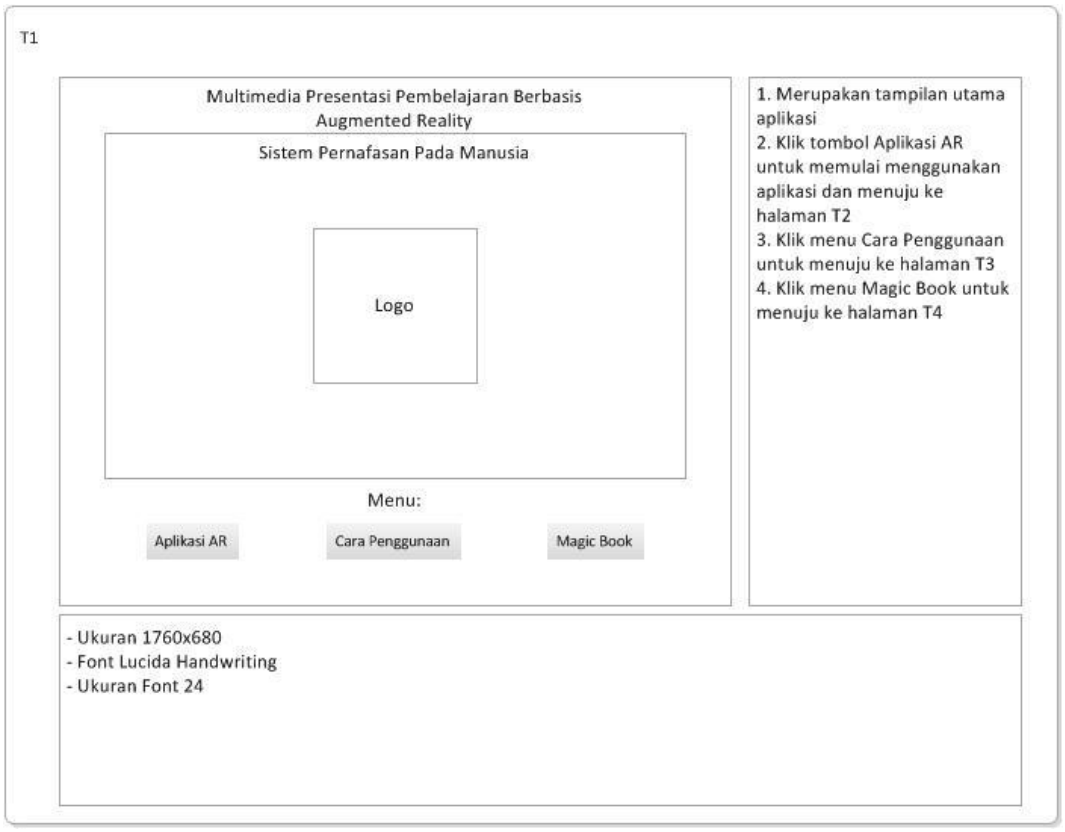
3.2 Perancangan

Perancangan akan dimulai setelah tahap analisis terhadap sistem selesai dilakukan. Perancangan dapat didefinisikan sebagai proses aplikasi berbagai teknik dan prinsip bagi tujuan pendefinisian suatu perangkat, suatu proses atau sistem dalam detail yang memadai untuk memungkinkan realisasi fisiknya.

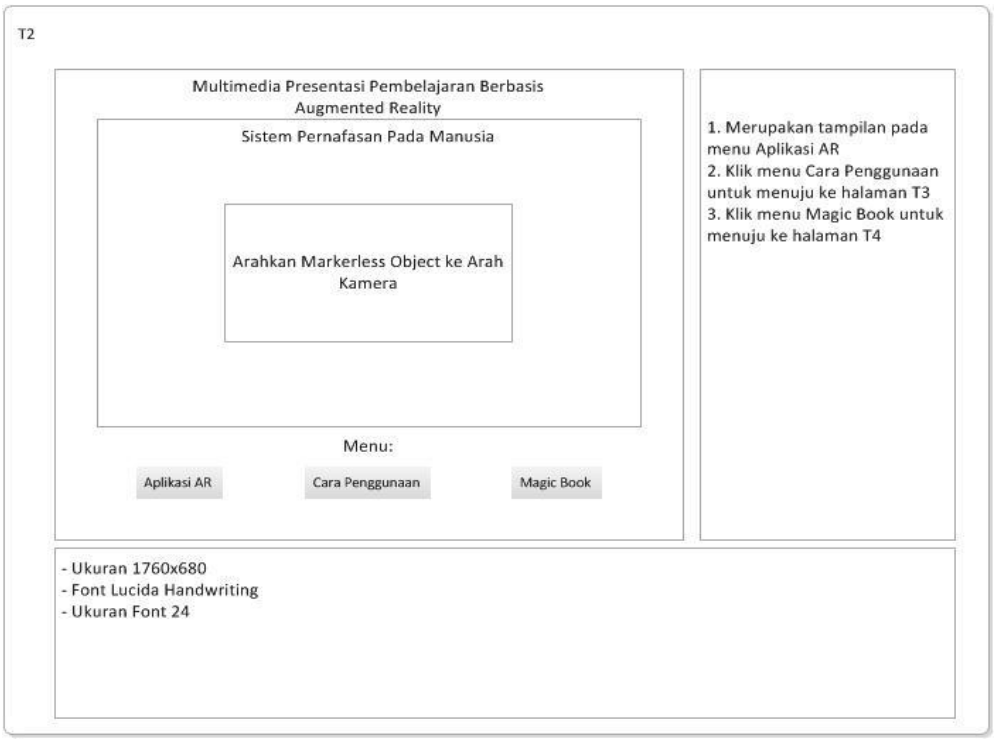
3.2.1 Perancangan Antarmuka

Desain yang dibuat berupa aplikasi berbasis desktop yang mudah untuk digunakan oleh pengguna, serta informasi yang dihasilkan aplikasi tersebut dapat dimengerti oleh pengguna.Perancangan antarmuka bertujuan untuk memberikan gambaran aplikasi yang dibuat. Obyek tiga dimensi merupakan obyek yang akan

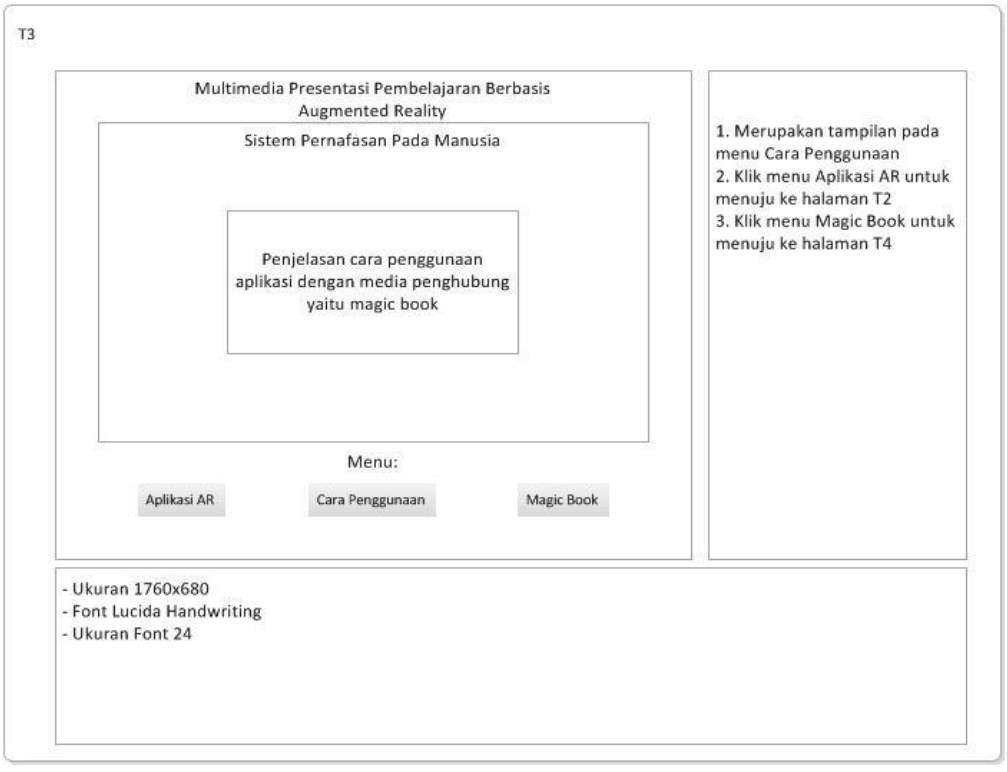
ditampilkan pada *marker* yang dapat dilihat oleh pengguna dengan bantuan *webcam*.



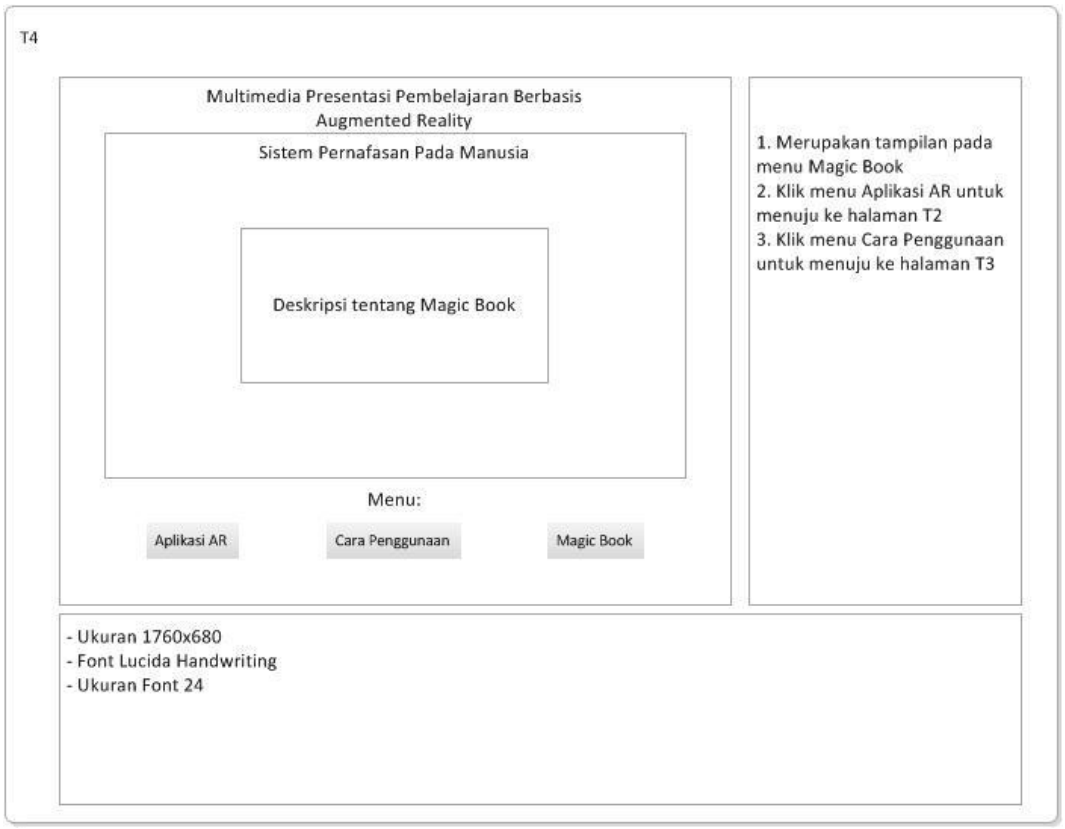
Gambar 3.18 Antar Muka Aplikasi AR Sistem pernafasan



Gambar 3.19 Tampilan Menu Aplikasi AR Siap Digunakan



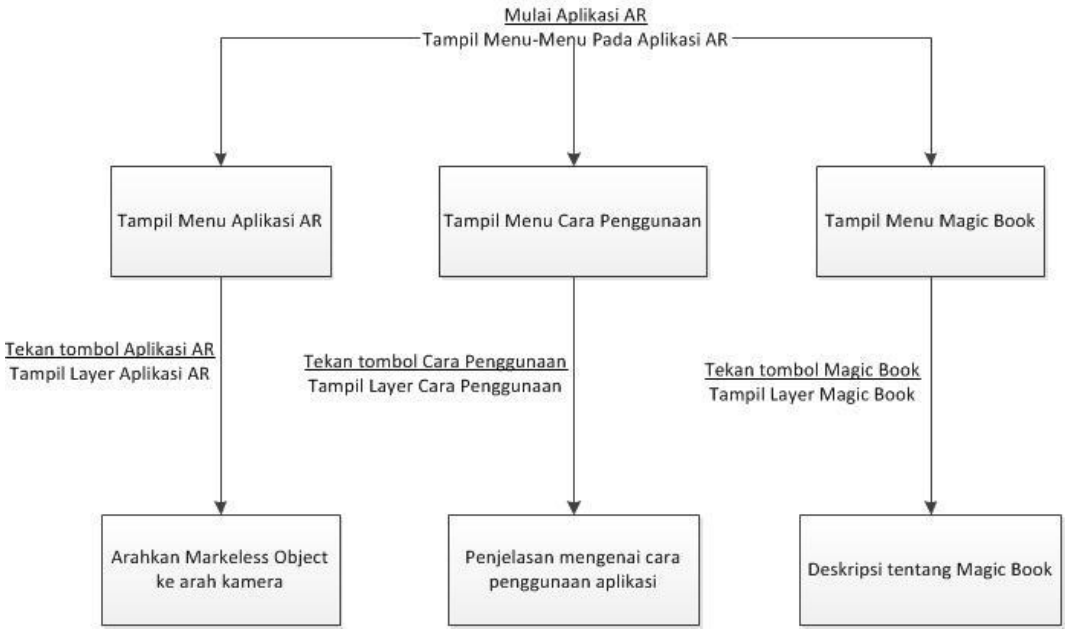
Gambar 3.20 Tampilan Menu Cara Penggunaan



Gambar 3.21 Tampilan Menu *Magic Book*

3.2.2 Perancangan Menu

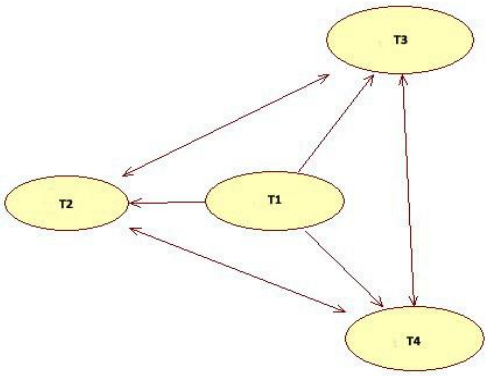
Menu pada aplikasi AR sistem pernafasan pada manusia ini adalah Aplikasi AR, Cara Penggunaan dan Magic Book. Berikut adalah perancangan menu pada aplikasi, seperti dijelaskan pada gambar 3.23 berikut:



Gambar 3.22 Perancangan menu aplikasi AR

3.2.3 Jaringan Semantik

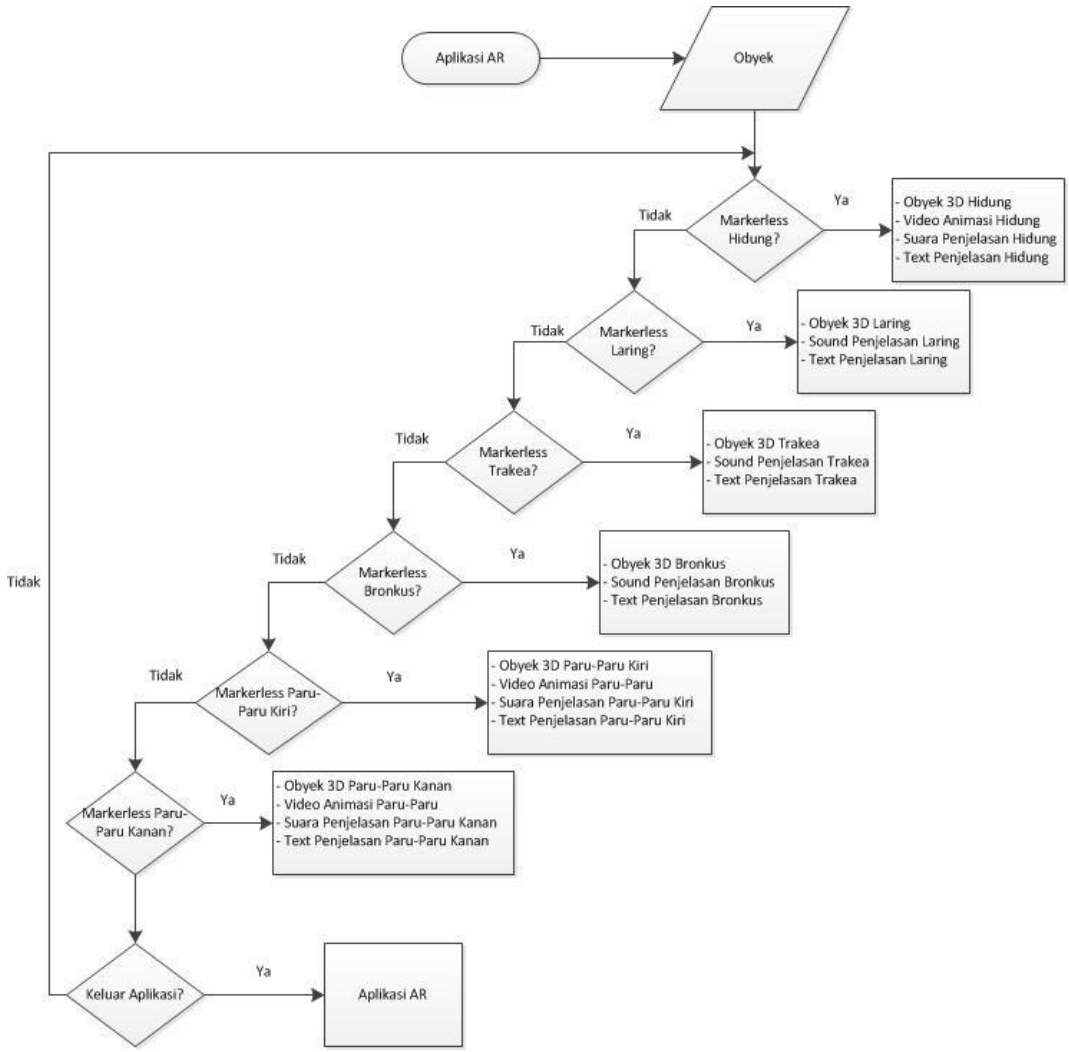
Jaringan semantik merupakan gambaran grafis yang menunjukkan antar obyek dan garis yang menggambarkan formasi antarobyek tersebut. Berikut ini adalah jaringan semantik pada aplikasi yang dibangun, seperti terlihat pada gambar 3.23.



Gambar 3.23 Jaringan Semantik Aplikasi AR Sistem pernafasan

3.2.4 Perancangan Method

Perancangan komponen method merupakan perancangan yang dibuat setelah perancangan antarmuka dan jaringan semantik. Perancangan ini berfungsi untuk mendeskripsikan method-method yang berada di dalam aplikasi Sistem Pernafasan Pada Manusia. Adapun method yang terdapat di dalam Sistem Pernafasan Pada Manusia yang dibangun adalah method aplikasi_ar(). Method Aplikasi_AR() akan dipanggil ketika user akan mengarahkan markerless pada menu Aplikasi AR, seperti terlihat pada gambar 3.24.



Gambar 3.24 *Flowchart* Aplikasi AR()