# Perancangan Kartu Nama dengan *Augmented Reality* sebagai Portofolio Digital

Demi Adidrana [(1)], A. Lumenta, ST., MT. [(2)], Brave A. Sugiarso, ST.,MT. [(3)], Virginia Tulenan, ST., MTI.[(4)]
(1)Mahasiswa (2)Pembimbing 1 (3)Pembimbing 2 (4)Pembimbing 3

demiadidrana@gmail.com[(1)] arie.lumenta@unsrat.ac.id [(2)] bravesugiarso@yahoo.com [(3)]
virginia.tulenan@unsrat.ac.id[(4)]

Jurusan Teknik Elektro-FT, UNSRAT, Manado-95115

***Abstract***

***Augmented reality is a term to combine the real world with the virtual world environment. Augmented reality became really popular today because it can be shown in real time, and it is really attractive. In general, augmented reality adds digital object which referred to certain objects in real world into reality. Therefore, this final project discuss about the creation of an application that aims to $_{present}$ a portfolio of art works collection using spatial disyplay technic with screen-based video see through displays method as an augmented reality display technic. Marker and motion detect as a reference object of the real world and portfolio animation as an object from the virtual world which are added to the real world to be more interesting.***

***The marker is used to call the portofolio animation to be shown at the monitor using base color recognition method which is red, green, and blue. While the motion detect is used to interact with the animation which is to move it left or right or to show the art works data of the portfolio using edge detection technic with canny operator.***

***Keyword: Augmented Reality, Edge Detection, Canny, Marker, Motion Detect, Portofolio.***

**Abstrak**

***Augmented reality* merupakan sebuah istilah untuk menggabungkan lingkungan dunia nyata dengan dunia virtual. *Augmented reality* menjadi sangat populer saat ini karena selain menarik, juga dapat di tampilkan secara *realtime*. Pada umumnya, *augmented reality* menambahkan objek digital pada realita dengan mengacu pada objek-objek tertentu pada dunia nyata, misalnya dengan menggunakan *marker, marker teksture (surface), face detection/recognition, motion detection,* dan *GPS & Digital Compass.* Oleh karena itu, pada tugas akhir ini di buatlah aplikasi yang bertujuan untuk mempresentasikan suatu portofolio yang berisi kumpulan karya lukisan dengan memanfaatkan konsep *augmented reality* yang menggunakan teknik *spatial display* dengan metode *screen-based video see through displays* sebagai teknik display *augmented reality*-nya. *Marker* dan *motion detect* adalah objek acuan pada dunia nyata dan animasi portofolio sebagai objek dari dunia virtual yang ditambahkan pada lingkup dunia nyata agar lebih menarik.**

***Marker* yang dibuat berfungsi untuk memanggil animasi portofolio muncul di layar komputer yang menggunakan teknik pengenalan warna dasar yaitu merah, hijau dan biru. Sedangkan *motion detect* berfungsi untuk berinteraksi dengan animasi portofolio, yaitu untuk menggerakan animasi ke kiri,kanan, dan untuk melihat data dari lukisan yang terdapat pada portofolio menggunakan teknik pendeteksian tepi dengan operator *canny*.**

Kata kunci : ***Augmented Reality*, Deteksi Tepi, *Canny, Marker, Motion Detect,* Portofolio**

## I. PENDAHULUAN

*Augmented Reality* (AR) merupakan sebuah istilah untuk benda-benda nyata dan maya di lingkungan nyata, berjalan secara interaktif dalam waktu nyata, dan terdapat integrasi antar benda dalam tiga dimensi, yaitu benda maya terintegrasi dalam dunia nyata. Penggabungan benda nyata dan maya dimungkinkan dengan teknologi tampilan yang sesuai, interaktivitas dimungkinkan melalui perangkat-perangkat input tertentu, dan integrasi yang baik memerlukan penjejakan yang efektif. Selain menambahkan benda maya dalam lingkungan nyata, realitas tertambah juga berpotensi menghilangkan benda-benda yang sudah ada. Menambah sebuah lapisan gambar maya dimungkinkan untuk menghilangkan atau menyembunyikan lingkungan nyata dari pandangan pengguna. Misalnya, untuk menyembunyikan sebuah meja dalam lingkungan nyata, perlu digambarkan lapisan representasi tembok dan lantai kosong yang diletakkan di atas gambar meja nyata, sehingga menutupi meja nyata dari pandangan pengguna(Ronald T. Azuma, 1997).

Pengaplikasian *Augmented reality* memiliki daya tarik tersendiri karena keunikannya, mulai dari perorangan hingga tingkat perusahaan pun tertarik menggunakan aplikasi *augmented reality*, seperti perusahaan Lego, General Electrics, yang mengaplikasikan AR untuk mempromosikan produknya. *Augmented reality* ini juga dikembangkan oleh perusahan-perusahaan permainan konsol modern untuk menambah daya tarik permainan, diantaranya adalah perusahaan Sony yang meluncurkan PSP Vita dengan fitur AR dalam permainannya. Tidak ketinggalan perusahaan pos di Amerika Serikat yaitu Priority Mail (United States Postal Service) pun menggunakan AR dalam salah satu produknya.

*Augmented reality* dapat juga dimanfaatkan untuk memvisualisasikan portofolio sebagai objek pada dunia maya yang digabungkan dalam kartu nama sebagai objek pada dunia nyata dengan menambahkan

*marker* sehingga portofolio dapat ditampilkan lebih menarik melalui kartu nama. Portofolio mempunyai arti yang luas dan berbeda dalam masing-masing bidang, namun secara umum portofolio merupakan suatu kumpulan dokumen atau hasil karya terbaik seseorang, kelompok, lembaga, organisasi, perusahaan atau sejenisnya yang bertujuan untuk mendokumentasikan perkembangan ataupun digunakan sebagai instrumen penilaian. Dalam dunia seni misalnya, bagi seorang seniman, model, ataupun arsitek yang mencari kerja, mereka senantiasa menyertakan "portofolio" dari hasil karya atau hasil kerja terdahulunya. Hasil kerja atau hasil karya tersebut adalah berupa karya foto, kliping majalah/koran, rancang bangun atau bukti-bukti lainnya.

Selain untuk menampilkan portofolio, untuk mempertegas definisi *Augmented reality* pada aplikasi maka dirancang pula pendeteksian gerakan (*motion detection*) yang menggunakan metode deteksi tepi dengan operator *Canny*. Pendeteksian gerakan tersebut akan digunakan sebagai sarana interaksi antar pengguna dan komputer.

Dari pemaparan diatas, maka untuk memanfaatkan keunggulan AR memvisualisasikan suatu portofolio dari kartu nama, diambillah judul "Perancangan Kartu Nama dengan *Augmented Reality* sebagai Portofolio Digital".

## II. LANDASAN TEORI

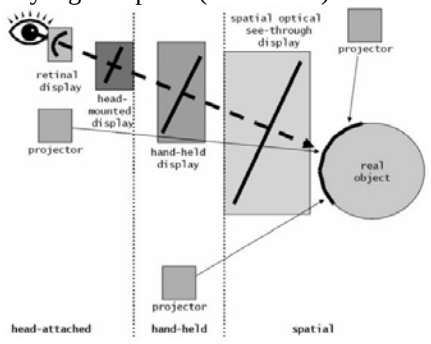
### A. *Augmented Reality*

*Augmented Reality* (AR) merupakan sebuah istilah untuk menggabungkan lingkungan dunia nyata dengan dunia virtual yang dibuat dengan komputer menggunakan bahasa pemrograman tertentu sehingga batas antar keduanya menjadi sangat tipis. Pada tahun 1997 Ronald Azuma mendefinisikan *augmented reality* sebagai sistem yang memiliki karakter sebagai berikut:

a. Menggabungkan lingkungan nyata dan virtual
b. Berjalan secara interaktif dalam waktu nyata
c. Integrasi dalam tiga dimensi (3D)

### B. *Teknik Display Augmented Reality*

Sistem *display* AR merupakan sistem manipulasi citra yang menggunakan seperangkat optik, elektronik, dan komponen mekanik untuk membentuk citra dalam jalur optik antara mata pengamat dan objek fisik yang akan digabungkan dengan teknik AR. Bergantung kepada optik yang digunakan, citra bisa dibentuk pada sebuah benda datar atau suatu bentuk permukaan yang kompleks (tidak datar).




Gambar 1. Pembentukan citra untuk *display augmented reality*

Gambar 1 mengilustrasikan kemungkinan citra akan dibentuk untuk mendukung AR, peletakan *display* bergantung dari pandangan pengguna dan objek, dan tipe citra seperti apa yang akan dihasilkan (*planar atau curved*).(Ronald T. Azuma, 1997)

Secara garis besarnya ada tiga teknik *display* AR, yaitu sebagai berikut:

1. *Head-Attached Display*
2. *Handheld Display*
3. *Spatial Display*

### C. *Spatial Display*

Dalam *Spatial Augmented Reality* (SAR), objek nyata digabungkan langsung dengan citra yang terintegrasi langsung ke lingkungan nyata. Contohnya, citra diproyeksikan ke lingkungan nyata menggunakan proyektor digital atau tergabung dengan lingkungan menggunakan panel *display*. Perbedaan utama pada SAR dibanding teknik *display* sebelumnya adalah *display*nya terpisah dengan pengguna. SAR memiliki kelebihan dari HMD dan *handheld*, sistem ini bisa digunakan oleh banyak orang pada waktu bersamaan tanpa perlu menggunakan suatu alat.(Ronald T. Azuma, 1997)

*a. Screen- Based Video See-Through Displays*
*Screen-Based* AR menggabungkan citra dan lingkungan nyata yang di tampilkan ke sebuah monitor, seperti yang ditunjukan pada gambar 2.

*b. Spatial Optical See-Through Displays*
Sistem ini menghasilkan citra yang ditampilkan langsung ke lingkungan nyata. Komponen yang penting dalam sistem ini meliputi *spatial optical combiners* (*planar atau curved beam combiners*), layar transparan atau hologram.

*c. Projection-Based Spatial Displays*
Sistem ini memproyeksikan citra secara langsung pada permukaan objek fisik daripada menampilkannya pada sebuah bidang pencitraan dalam penglihatan pengguna. Sistem ini menggunakan banyak proyektor yang digunakan untuk meningkatkan wilayah tampilan serta meningkatkan kualitas citra.

### D. *Pengertian Citra Digital*

Secara umum, pengolahan citra digital menunjuk pada pemrosesan gambar 2 dimensi menggunakan komputer. Dalam konteks yang lebih luas, pengolahaan citra digital mengacu pada pemrosesan setiap data 2 dimensi.

Gambar 2 Contoh *Screen-Based Video See-Through Displays*

Citra digital merupakan sebuah larik (*array*) yang berisi nilai-nilai *real* maupun komplek yang direpresentasikan dengan deretan bit tertentu.

Suatu Citra dapat di definisikan sebagai fungsi f(x,y) berukuran M baris dan N kolom, dengan x dan y adalah koordinat spasial, dan amplitude f di titik koordinat (x,y) dinamakan intensitas atau tingkat keabuan dari citra pada titik tersebut. Apabila nilai x,y, dan amplitude f secara keseluruhan berhingga (*finite*) dan bernilai diskrit maka dapat dikatakan bahwa citra tersebut adalah citra digital(Darma Putra, 2004). Gambar 3 menunjukan posisi koordinat citra digital.

Citra digital dapat ditulis dalam bentuk matrik sebagai berikut (Darma Putra, 2004)

$$f(x,y) \approx \begin{bmatrix} f(0,0) & f(0,1) & \dots & f(0,M-1) \\ f(1,0) & f(1,1) & \dots & f(1,M-1) \\ \vdots & \vdots & \vdots & \vdots \\ f(N-1,0) & f(N-1,1) & \dots & f(N-1,M-1) \end{bmatrix} \quad (1)$$

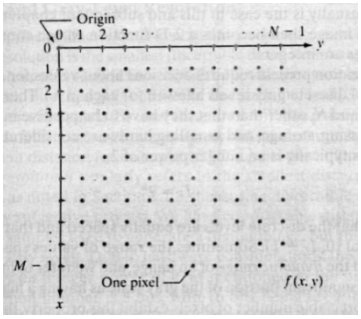
Berdasarkan jenisnya, citra digital dapat dibagi menjadi 3 (Sutoyo, 2009), yaitu:

1. Citra Biner (Monokrom)

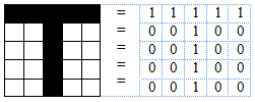
Memiliki 2 buah warna, yaitu hitam dan putih. Warna hitam bernilai 1 dan warna putih bernilai 0. Untuk menyimpan kedua warna ini dibutuhkan 1 bit di memori. Contoh dari susunan piksel pada citra monokrom adalah sebagai berikut (gambar 4).

2. Citra *Grayscale* (skala keabuan)

Citra *grayscale* mempunyai kemungkinan warna hitam untuk nilai minimal dan warna putih untuk nilai maksimal. Banyaknya warna tergantung pada jumlah bit yang disediakan di memori untuk menampung kebutuhan warna tersebut. Semakin besar jumlah bit warna yang disediakan di memori, maka semakin halus gradasi warna yang terbentuk (gambar 5).




Gambar 3 Koordinat citra digital




Gambar 4 Contoh susunan *pixel* citra *monokrom*

Gambar 5 Citra *grayscale*

3. Citra Warna (*true color*)

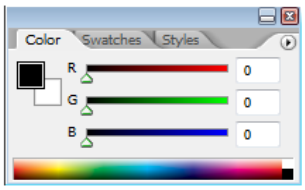
Setiap piksel pada citra warna mewakili warna yang merupakan kombinasi tiga warna dasar, yaitu merah, hijau, dan biru (RGB = *Red*, *Green*, *Blue*). Setiap warna dasar menggunakan penyimpanan 8 bit = 1 *byte* (nilai maksimum 255 warna), jadi satu piksel pada citra warna diwakili oleh 3 *byte*.(gambar 6)

*E.* Deteksi Tepi (*Edge Detection*)

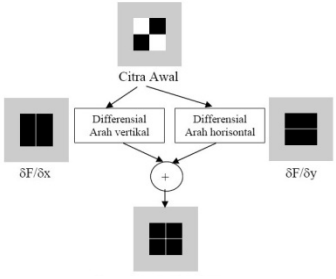
Tepian dari suatu citra mengandung informasi penting dari citra bersangkutan. Tepian citra dapat merepresentasikan objek – objek yang terkandung dalam citra tersebut, bentuk, dan ukurannya serta terkadang juga informasi tentang teksturnya. Tepian citra adalah posisi dimana intensitas *pixel* dari citra berubah dari nilai rendah ke nilai tinggi atau sebaliknya. Deteksi tepi umumnya adalah langkah awal melakukan segmentasi citra (Darma Putra, 2004).

Deteksi tepi *(edge detection)* adalah operasi yang dijalankan untuk mendeteksi garis tepi (*edges*) yang membatasi dua wilayah citra *homogeny* yang memiliki tingkat kecerahan yang berbeda (Pitas 1993). Deteksi tepi pada suatu citra adalah suatu proses yang menghasilkan tepi-tepi dari obyek-obyek citra, tujuannya adalah :

a. Untuk menandai bagian yang menjadi detail citra
b. Untuk memperbaiki detail dari citra yang kabur, yang terjadi karena *error* atau adanya efek dari proses akuisisi citra.
c. Serta untuk mengubah citra 2D menjadi bentuk kurva suatu titik (x,y) dikatakan sebagai tepi (*edge*) dari suatu citra bila titik tersebut mempunyai perbedaan yang tinggi dengan tetangganya. Gambar 7 berikut ini menggambarkan bagaimana tepi suatu gambar di peroleh.




Gambar 6 Citra warna




Gambar 7 Proses deteksi tepi

Ada dua metode untuk dapat mendeteksi tepi yaitu (wasista, 2009):

1. Metode *First-Order Derivative Edge Detection*
2. Metode *Second-Order Derivative Edge Detection*

*F.* Operator *Canny*

Deteksi tepi Canny dapat mendeteksi tepian yang sebenarnya dengan tingkat kesalahan minimum. Dengan kata lain, operator Canny di desain untuk menghasilkan citra tepian yang optimal. Berikut adalah langkah-langkah dalam melakukan deteksi tepi Canny (Darma Putra, 2004).

1. Menghilangkan derau yang ada pada citra dengan mengimplementasikan tapis *Gaussian*. Proses ini akan menghasilkan citra yang tampak sedikit buram. Hal ini dimaksudkan untuk mendapatkan tepian citra yang sebenarnya. Bila tidak dilakukan maka garis-garis halus juga akan di deteksi sebagai tepian. Berikut ini adalah salah satu contoh tapis *Gaussian* dengan σ = 1,4.

$$B = \frac{1}{115}\begin{bmatrix} 2 & 4 & 5 & 5 & 2 \\ 4 & 9 & 12 & 9 & 4 \\ 5 & 12 & 15 & 12 & 5 \\ 4 & 9 & 12 & 9 & 4 \\ 2 & 4 & 5 & 4 & 2 \end{bmatrix} \quad (2)$$

2. Melakukan deteksi tepi dengan salah satu deteksi tepi turunan pertama dengan melakukan pencarian secara horizontal (Gx) dan vertikal (Gy).
3. Menentukan arah tepian yang ditemukan dengan menggunakan rumus sebagai berikut.

$$G = \sqrt{Gx^2 + Gy^2} \quad (3)$$

$$\theta = arctan\left(\frac{Gy}{Gy}\right) \quad (4)$$

   Selanjutnya membagi ke dalam 4 warna sehingga garis arah berbeda dan memiliki warna berbeda. Pembagiannya adalah 0 – 22,5 dan 157,5 – 180 derajat berwarna kuning, 22,5 – 67,5 berwarna hijau, dan derajat 67,5 – 157,5 berwarna merah.
4. Memperkecil garis tepi yang muncul dengan menerapkan *nonmaximum suppression* sehingga menghasilkan garis tepian yang lebih ramping.
5. Langkah terakhir adalah berinisiasi dengan menerapkan dua buah *tresholding*.

*G. Adobe Flash*

*Adobe Flash* merupakan sebuah program yang ditujukan kepada para desainer atau programer yang bertujuan merancang animasi untuk pembuatan sebuah halaman web, pembuatan game interaktif, presentasi untuk tujuan bisnis, proses pembelajaran, pembuatan film kartun, dan dapat digunakan untuk membangun sebuah aplikasi yang bernilai tinggi serta tujuan – tujuan lain yang lebih spesifik lagi. Teknologi *flash* menjadi solusi bagi penyebar informasi atau pembangunan aplikasi untuk disebarkan ke khalayak ramai sehingga menjadi teknologi yang popular dan berkembang dengan pesat. *Flash* dapat dilihat dari dua aspek, yaitu:

1. *Flash* sebagai *software*.
   *Adobe Flash* sebagai *software* pembuat atau pembangun aplikasi, system informasi, dan pembuat animasi.
2. *Flash* sebagai teknologi.
   Sekarang ini hampir semua *browser* serta sebagian peralatan elektronik sudah terinstal *Flash Player* untuk dapat menjalankan animasi.

*Flash* adalah program animasi yang berbasis *vector* yang dapat menghasilkan file yang berukuran kecil sehingga mudah di akses, *Flash* dilengkapi dengan *tool – tool* untuk membuat gambar yang kemudian akan dibuat animasi atau dijalankan dengan *script*-nya (*ActionScript*).

## III. METODOLOGI PENELITIAN

*A. Tempat dan Waktu Penelitian*

Dalam pelaksanaan tugas akhir ini penulis mengambil tempat penelitian pada Ruang Laboratorium Sistem Komputer (LSK), Jurusan Teknik Elektro, Fakultas Teknik Universitas Sam Ratulangi (UNSRAT), dan rumah penulis.

*B. Bahan dan Peralatan*

Dalam pembuatan tugas akhir ini, penulis menggunakan peralatan dan program yang disesuaikan dengan kebutuhan dalam pembuatan aplikasi dengan *augmented reality*. Secara lebih spesifik, peralatan dan program yang digunakan dirinci sebagai berikut:

1. Spesifikasi laptop yang digunakan yaitu:
   a. Sistem operasi *Windows XP* 32-bit.
   b. *Processor* Intel Core i3-380
   c. RAM DDR2 2 GB
   d. *Harddisk* 320 GB.
   e. *Webcam* eksternal Logitech 720HD
   f. *Webcam* internal 320 DPI
2. Perangkat lunak
   a. *Windows 7*, berfungsi sebagai sistem operasi
   b. *Adobe Flash CS5*, berfungsi sebagai *IDE*
   c. *Adobe Flash Builder*, berfungsi sebagai *script editor*
   d. *Flash player 10-11*, berfungsi sebagai *debugger*
   e. *Mozilla Firefox*, berfungsi sebagai *web browser*

*C. Prosedur Penelitian*

Prosedur yang dilakukan dalam membuat perancangan kartu nama dengan *augmented reality* adalah sebagai berikut:

1. Melakukan studi literatur dan mencari referensi yang berhubungan dengan *augmented reality*, *actionscript* dan *flash*, dan pengolahan citra digital.
2. Melakukan pengambilan data berupa biodata dan portofolio dari pelukis.
3. Melakukan pembuatan animasi.
4. Melakukan pembuatan deteksi *marker*.
5. Melakukan perancangan *marker*.
6. Melakukan pembuatan *motion detect* dengan *actionscript*.
7. Merancang *interface* aplikasi *augmented reality.*

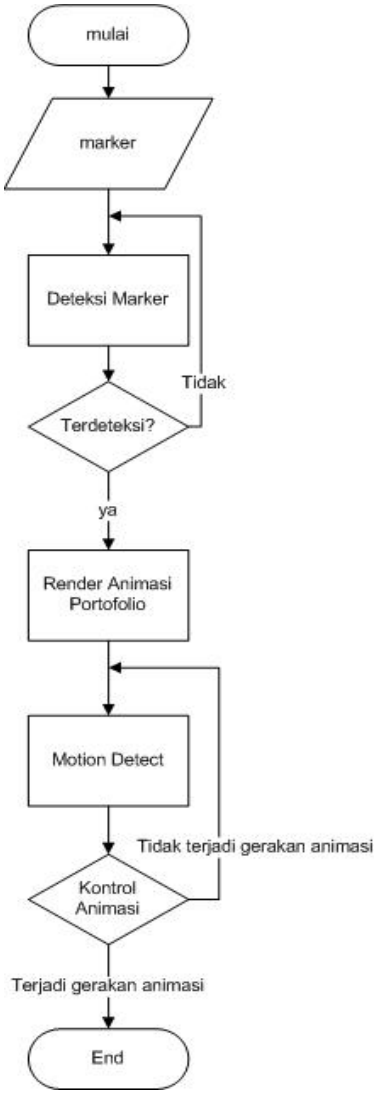
8. Melakukan pengujian dari aplikasi *augmented reality*.
9. Mengimplementasikan aplikasi yang telah dibuat.

*D. Perancangan Program*

Perancangan program aplikasi ini dilakukan dengan menerapkan pendeteksian warna RGB *(Red-Green-Blue)* untuk mendeteksi *marker* dengan pola warna tersebut yang akan di jadikan sebagai *input* untuk aplikasi *augmented reality*. Pada aplikasi *augmented reality*, objek animasi akan di *render* jika *marker* telah terdeteksi lalu akan diterapkan deteksi tepi dengan operasi canny untuk deteksi *motion* untuk interaksi antara pengguna dan objek animasi. Secara terperinci, perancangan dan aplikasi akan dibagi menjadi beberapa bagian, dengan *flowchart* seperti pada gambar 8.

*E. Mendeteksi Marker*

Proses ini dimulai dengan mengakses kamera pada komputer (*plug-and-play webcam, embedded webcam*, dan sebagainya), kemudian melakukan kalibrasi singkat ukuran *video* dan kualitasnya (*fps*), dan menangkap inputan berupa gambar *live* yang terhampar di hadapan kamera.




Gambar 8 Flowchart Program

Pada tahap akses ini, *user* akan menghadapkan *marker* pada kamera.

Kamera dan program akan menerima gambar dari *video stream* kamera. Hasil gambar dari input *video stream* ini akan dikalkulasikan setiap pixelnya. Kemudian,program akan mencari pola warna pada *marker*. Setelah menemukan pola warna *RGB* pada *marker*. Program akan mengkalkulasi pola warna berdasarkan posisi dari ketiga warna (merah di atas,biru di samping, hijau di samping) tersebut dan membandingkan jumlah pixel dari ketiga warna tersebut. *Marker* akan terbaca jika orientasi warna merah berada di atas, jika orientasi tidak sesuai maka *marker* tidak akan terbaca, jika *marker* telah terbaca maka proses pendeteksian akan dihentikan untuk meminimalisir penggunaan memori komputer. Berikut adalah logika algoritmanya.

```
P = position, PTop = top position, PL = left
position, PR = right position
For each pixel (Px) in the image
If (P = 1)
        PT.RED > PL.RED  and PR.RED
Else If (P = 0)
        PL.GREEN > PT.GREEN and PR.GREEN
        PR.BLUE > PT.BLUE and PR.BLUE
Calculate the score of the pixel
End if
End for
```

*F. Render Animasi*

Proses ini dimulai ketika aplikasi telah mendeteksi *marker* yaitu aplikasi akan mengambil gambar dari lokasi sumber gambar dan menyimpannya dalam *array*. Gambar yang telah tersimpan dalam *array* tersebut lalu akan ditempatkan pada *movieclip* dengan meng-iterasi jumlah *movieclip* sesuai dengan jumlah gambar. Jika jumlah *movieclip* sama dengan jumlah gambar maka iterasi akan di hentikan lalu akan diposisikan dengan bentuk lingkaran 3 dimensi dimana *movieclip* pertama akan berada paling depan, dan yang terakhir paling belakang.

Ketika proses *render* animasi dipanggil, secara bersamaan pula akan dilakukan proses pendeteksian gerakan.

*G. Mendeteksi Gerakan*

Setelah *marker* terdeteksi maka proses ini akan dimulai bersamaan dengan proses *render* animasi. Tujuan dari proses mendeteksi gerakan adalah untuk sarana interaksi objek animasi dengan *user* seperti menggerakan animasi ke kiri dan kanan, dan menampilkan detail dari gambar lukisan.

Pada proses deteksi gerakan ini program akan mengambil input dari *video stream* yang di tangkap kamera dan menyimpannya dalam variable dengan tipe bitmapdata, mengkonversinya ke dalam resolusi yang lebih kecil dan tidak akan ditampilkan ke layar komputer, ini dilakukan karena proses terjadi hampir bersamaan secara *live* dengan tampilan dari *video stream* dan untuk mengkalkulasi setiap pixel dibutuhkan *resource* memori makin besar jika ukuran dari *video stream* makin besar pula.

Pertama – tama akan dilakukan deteksi tepi dengan operator canny. Mengapa menggunakan

operator ini karena dapat mendeteksi tepian yang sebenarnya dengan tingkat kesalahan minimum (Darma Putra,2010). Dengan kata lain, operator canny di desain untuk menghasilkan citra tepian yang optimal. Berikut adalah langkah-langkah dalam melakukan deteksi tepi canny yang digunakan dalam program.

1. Menghilangkan derau yang ada pada citra dengan mengimplementasikan tapis *Gaussian*. Proses ini akan menghasilkan citra yang tampak sedikit buram. Hal ini dimaksudkan untuk mendapatkan tepian citra yang sebenarnya. Bila tidak dilakukan maka garis-garis halus juga akan di deteksi sebagai tepian. Berikut ini adalah tapis *Gaussian* yang digunakan dengan $\sigma = 1{,}4$ dengan matriks pada rumus (2).

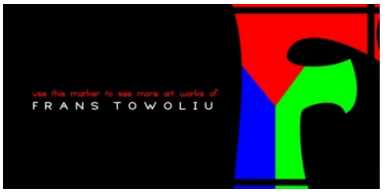
2. Melakukan deteksi tepi dengan operator *sobel*, yaitu dengan mengkalkulasi dan men-iterasi nilai input pixel dari setiap frame dengan matriks operator *sobel*.

| GX | GY | | pixel | |
|---|---|---|---|---|
| 1 -2 -1 | -1 0 1 | atasKiri | atas | atasKanan |
| 0 0 0 | -2 0 2 | Kiri | ----- | Kanan |
| 1 2 1 | -1 0 1 | bawahKiri | bawah | bawahKanan |

3. Menentukan arah tepian yang ditemukan dengan menggunakan rumus (2) dan (3).
4. Memperkecil garis tepi yang muncul dengan menerapkan *nonmaximum suppression* sehingga menghasilkan garis tepian yang lebih ramping.
5. Langkah terakhir adalah menyimpan nilai pixel yang telah dikalkulasikan dengan variable baru. Dan menerapkan *tresholding* antara nilai *input* dan nilai hasil kalkulasi.

Untuk mendapatkan *motion detect* menggunakan metode *difference blend mode*, yang pada dasarnya adalah membandingkan warna merah, hijau, dan biru dari masing-masing *pixel* per-*frame* (*frame* sebelum, *frame* sesudah) maka didapatkan nilai selisih dari kedua *frame* tersebut (*blend frame*). Untuk mendapatkan warna hitam yang lebih akurat maka *frame* yang di proses adalah output dari deteksi tepi.


Gambar 9 Bagian depan kartu nama




Gambar 10 Bagian belakang kartu nama yang tercetak marker

## IV. PENGUJIAN DAN PEMBAHASAN

### A. *Pembuatan Kartu Nama dan Marker*

Dalam pembuatan kartu nama dan *marker* untuk aplikasi *augmented reality* ini di desain menggunakan program *coreldraw*. *Marker* ini akan di cetak pada bagian belakang kartu nama yang dapat di lihat pada gambar di bawah ini (gambar 9 dan 10).

Dapat diperhatikan pada gambar 10 yang digunakan sebagai *marker* adalah gambar “F” yang memiliki warna sesuai spesifikasi dari aplikasi *augmented reality* yang di bangun.

### B. *Akses Kamera*

Untuk pemanggilan kamera, dibutuhkan koneksi antara program dan kamera. Berikut adalah pembuatan koneksi pada kamera.

```
// init camera

private function initCamera():void{
    // camera
    camera=Camera.getCamera();
    camera.setMode(640,480, 30, false);
```

Resolusi kamera diatur tidak terlalu besar agar tidak banyak membebani memori komputer.

### C. *Pembuatan Deteksi Marker*

Pembuatan deteksi *marker* dimulai dengan mengambil nilai *pixel* dari kamera dan menyimpannya dalam variabel *vid_marker*.

```
// vid_marker deteksi marker
vid_marker=new Video(camera.width,camera.height);
vid_marker.attachCamera(camera);
vid_marker.scaleX = -1;
vid_marker.x = 640;
vid_marker.y = 0;

addChild(vid_marker);
```

Secara *default flash* menampilkan *video stream* kamera secara *mirrored* (gambar 11) maka nilai besar X (*vid_marker.scaleX*) dirubah menjadi -1, dan posisi x dari *vid_marker* dikembalikan menjadi 640, sedangkan y-nya = 0 agar posisi dari *vid_marker* tidak berada diluar jangkauan *stage* (gambar 12).

Gambar 11 *Output* kamera sebelum di proses

Gambar 12 *Output* vid_marker

Untuk mendeteksi *marker* dibuat seuai algoritma pada bab sebelumnya. Berikut adalah potongan *script* implementasi algoritma tersebut.

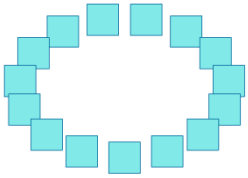
*D. Pembuatan Animasi*

Animasi adalah unsur *augmented reality* yang akan di tampilkan pada aplikasi. Animasi merupakan objek maya yang ditambahkan pada dunia nyata yang tampil di layar komputer bersamaan dengan tampilan yang ditangkap kamera. Animasi yang dibuat berisi dari kumpulan *container* yang berisi gambar dan disusun berurutan menjadi lingkaran 3 dimensi. (gambar 13 dan 14)

*E. Pembuatan Motion Detect*

*Motion Detect* dibuat melalui 3 tahap utama yaitu,

1. Deklarasi *vid_motion*
   *Vid_motion* di deklarasikan sebagai variabel dari *class video* dari *actionscript* 3, variabel ini akan menyimpan nilai dari lebar dan tinggi *output* kamera. Nilai tersebut selanjutnya akan di konversi ke resolusi lebih kecil untuk mengurangi *resource* memori komputer.
   Sama hal-nya dengan variabel *vid_marker*, *vid_motion* juga menerapkan nilai -1 pada *vid_motion.scaleX*, ini dilakukan untuk merubah *output* kamera yang secara *default* ter-*mirrored*. Untuk memperhalus gambar dan mengurangi derau diterapkan *filter blur* dengan menggunakan *class filters* dari *actionscript* 3.
2. *Edge Detection*
   *Input* untuk deteksi tepi diambil dari nilai lebar dan tinggi *output vid_motion* dan disimpan dalam variabel *inputBM* yang di deklarasikan sebagai *bitmapdata*. Sesuai dengan ketentuan deteksi tepi operator canny, *inputBM* akan mengimplementasikan tapis *gaussian* untuk mengurangi derau yang menghasilkan citra tampak buram. Selanjutnya *inputBM* yang telah di beri tapis *gaussian* akan di iterasi masing – masing nilai matrix pixelnya dengan matrix operator sobel yang dilakukan dengan pencarian secara horisontal (Gx) dan secara vertikal (Gy). Nilai dari hasil iterasi deteksi tepi akan disimpan dalam variabel baru yang akan di tampilkan sebagai *display list*.



Gambar 13 *Container* sebelum load gambar

Gambar 14 *Container* telah men-*load* gambar

3. *Difference Blend Mode*
   Konsep dari deteksi gerakan ini adalah membandingkan 2 *frame* secara berturut-turut, dimana perbedaan nilai pixel dari masing-masing *frame* menandakan adanya suatu pergerakan menggunakan metode *difference blend mode*. Untuk mengimplementasikannya di deklarasikan 2 variabel *bitmap*: ‘ sebelum dan sesudah’. *Input* untuk *motion detect* adalah *output* dari deteksi tepi yang akan di *threshold* per-*frame*nya. Metode *threshold* bekerja dengan memeriksa dan membandingkan setiap nilai dari pixel *frame*.

*F. Pembuatan Kontrol*

Untuk membuat animasi bergerak dengan *motion detect* maka perlu dibuat control untuk animasi tersebut. Konsepnya adalah jika gerakan yang ditangkap menuju ke sebelah kiri layar maka animasi akan bergerak ke kiri sedangkan untuk ke kanan adalah sebaliknya. Sedangkan untuk menandakan adanya pergerakan dibuat *sprite* dengan bentuk lingkaran yang berfungsi sebagai *pointer* dan untuk men*trigger* agar animasi bergerak dibuat *movieclip* pada samping kiri,kanan,atas kanan, dan atas kiri dan tengah (gambar 15).

*G. Pengujian Deteksi Marker*

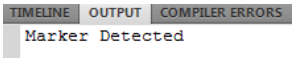
Pengujian ini dilakukan untuk mengetahui apakah *marker* terdeteksi, *output* dari pengujian ini akan muncul pada menu *output* dari *flash* sesuai *trace* yang di atur. (gambar 16 dan 17)

*H. Pengujian Sensitivitas Gerakan*

Karena deteksi yang dilakukan pada gerakan, maka semua gerakan yang di tangkap *webcam* akan mempengaruhi gerakan dari *pointer* dan dari keseluruhan gerakan yang di deteksi dipadukan menjadi 1(satu) gerakan saja. Uji coba yang di lakukan terbatas hanya pada gerakan 1 orang dan beberapa gerakan yang mungkin terjadi seperti yang digambarkan pada gambar 18. Pada gambar tersebut terjadi 4 gerakan berbeda yang mempengaruhi pergerakan *pointer*.

Gambar 15 Output kontrol

Gambar 16 Uji coba *marker* orientasi = 0




Gambar 17 Uji coba *marker* terdeteksi

Hasil dari tangkapan *webcam* juga sangat dipengaruhi oleh intensitas cahaya ruangan. Misalnya pada saat pengujian terdapat pantulan cahaya yang sangat terang, maka pada daerah tersebut akan sangat sensitif pada gerakan yang ada.

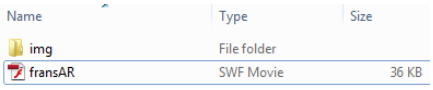
## *I. Implementasi Sistem*

Aplikasi yang telah di uji selanjutnya akan di implementasikan dengan penggabungan dari fungsi-fungsi yang telah dijelaskan sebelumnya dimulai dari pemanggilan kamera,pendeteksian *marker*, *render* animasi, dan *motion detect* sehingga menjadi satu kesatuan aplikasi *augmented reality*.

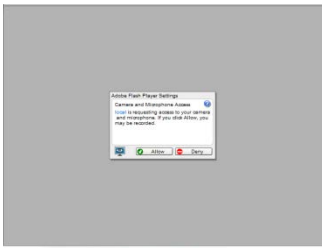
Aplikasi yang telah di *compile* oleh flash akan berekstensi *SWF* dengan nama fransAR.swf. File ini harus di simpan dalam satu struktur folder dimana isi folder tersebut adalah file foto dari portofolio (gambar 20)**.**

Tampilan pertama saat aplikasi dibuka adalah *flash* akan meminta izin *user* untuk mengizinkan atau tidak untuk menggunakan *webcam* (*lihat gambar 21*), lalu akan muncul tampilan untuk pendeteksian *marker* (*lihat gambar 22*). Pada tampilan ini *marker* diarahkan ke *webcam*.

Gambar 18 Contoh gerakan yang mungkin terjadi




Gambar 20 Struktur folder



Gambar 21 Tampilan awal pemanggilan *webcam*

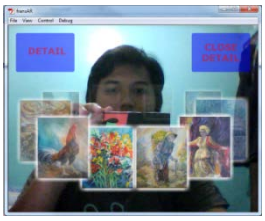
Gambar 22 *Marker* di arahkan ke *webcam*

Setelah *marker* di arahkan dengan orientasi yang tepat maka objek-objek animasi portofolio akan ditampilkan di depan *user* (gambar 23).

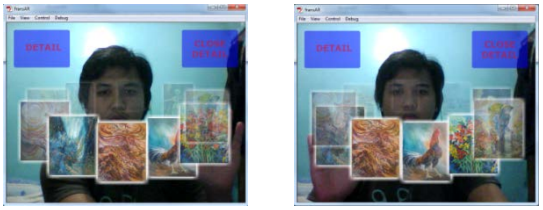
Pada saat objek telah di *load* sempurna maka *marker* tidak perlu di arahkan lagi karena fungsi untuk mendeteksi *marker* telah di hentikan bersamaan dengan terdeteksinya *marker* tersebut. Animasi pun dapat di gerakan berputar sesuai dengan gerakan ke kanan layar ataupun ke kiri layar (gambar 24).

Untuk melihat detail data dari portofolio *user* cukup menggerakkan tangan ke tengah dari animasi sehingga akan menghentikan gerakan animasi, lalu menggerakkan tangan ke tombol detail (gambar 25). Sedangkan jika akan menghentikan putaran animasi saja dapat menggerakan tangan ke tengah animasi (gambar 26).

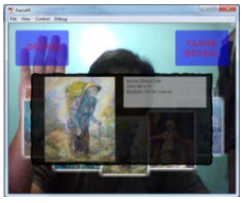
Setelah melihat data dari portofolio *user* dapat menutupnya dengan mengarahkan tangan ke tombol *close*, lalu *user* dapat mengulangi tahapan sebelumnya untuk melihat data portofolio yang lainnya (gambar 27).




Gambar 23 *Marker* terdeteksi dan menampilkan objek




Gambar 24 Menggerakan Animasi ke kanan dan kiri



Gambar 25 Melihat data portofolio


Gambar 26 Menghentikan animasi


Gambar 27 Menutup jendela detail

## V. KESIMPULAN

Berdasarkan hasil penelitian yang dilakukan, diperoleh beberapa kesimpulan sebagai berikut:

1. Aplikasi *augmented reality* yang dibuat menggunakan teknik *spatial display* (*screen-based video see-through displays*) yang memerlukan komputer, *webcam*, dan *marker*.
2. Karena keterbatasan dari penangkapan citra oleh *webcam*, maka intensitas cahaya sangat berpengaruh.
3. Interaksi yang dapat ditangani aplikasi hanyalah untuk menggerakan objek ke arah kiri dan kanan dan membuka jendela baru (*detail view*).
4. Karena deteksi yang dilakukan terbatas pada tepi (*edge detect)* dan gerakan (*motion detect*), maka setiap gerakan akan terdeteksi akan mampu men-*trigger* interaksi dengan objek di layar.

## DAFTAR PUSTAKA


[1] Adobe Systems Incorporated, "ActionScript 3.0 Reference for the Adobe FlashPlatform", tersedia di: http://help.adobe.com/en_US/FlashPlatform/reference/actionscript/3/flash/evens/Event.html, diakses: 27 Oktober 2012.

[2] Adobe Systems Incorporated, *"Animation Learning Guide for Flash"*, tersedia di: http://www.adobe.com/devnet/flash/learning_guide/animation.html, diakses: 29 Oktober 2012.

[3] R. Akbar, *"Augmented Reality – Pengenalan (part-1)"*, tersedia di: http://augindonesia.org/augmented-reality-pengenalan-part-1/, diakses: 17 Agustus 2012.

[4] R. Akbar, *"Augmented Reality – Perkembangan (part-2)"*, tersedia di: http://augindonesia.org/augmented-reality-perkembangan-part-2/, diakses: 17 Agustus 2012.

[5] R. Akbar, *"Augmented Reality – Objek Acuan (part-3)*" tersedia di: http://augindonesia.org/augmented-reality-object-acuan-part-3/, diakses: 17 Agustus 2012.

[6] R. Braunstein, H.W. Mims, & J.N. Joshua, *"ActionScript 3.0 Bible"*, Canada: Willey Publishing, 2008.

[7] A. Fathoni, *"Metodologi Penelitian & Teknik Penyusunan Skripsi"*, Jakarta: Rineka Cipta, 2011.

[8] J. Lott, D. Schall, & K. Peters, *"ActionScript 3.0 Cookbook"*, Sebastopol: O'Reilly, 2006.

[9] Madcoms, *"Kupas Tuntas Adobe Flash Profesional CS5"*, Yogyakarta: Andi Offset, 2011.

[10] T. McCauley, *"Getting Started with ActionScript 3.0 in Adobe Flash CS3"*, tersedia di: http://www.senocular.com/flash/tutorials/as3withflashcs3/, diakses: 2 November 2012.

[11] C. Moock, *"Essential ActionScript 3.0"*, Sebastopol: O'Reilly, 2007.

[12] K. Peters, *"Fondation ActionScript 3.0 Animation: Making Things Move!*", United States of America: Springer, 2007.

[13] K. Peters, *"AdvancED ActionScript 3.0 Animation*", United States of America: Springer, 2009.

[14] G. Pranowo, *"Kreasi Animasi Interaktif dengan ActionScript 3.0 pada Flash CS5"*, Yogyakarta: Andi Offset, 2011

[15] D. Putra, *"Pengolahan Citra Digital"*, Yogyakarta: Andi Offset, 2011.

[16] S.A. Rivello, *"Augmented Reality Using Webcam and Flash*", tersedia di : http://www.adobe.com/devnet/flash/articles/augmented_reality html. diakses: 25 Agustus 2012.

[17] W. Sanders, C. Chandima, *"ActionScript 3.0 Design Patterns*", Sebastopol: O'Reilly, 2007.