MULTI MARKER AUGMENTED REALITY UNTUK APLIKASI *MAGIC BOOK*

Akhmad Afissunani, Akuwan saleh, M. Hasbi Assidiqi
Politeknik Elektronika Negeri Surabaya
Institut Teknologi Sepuluh Nopember, Kampus ITS, Surabaya 60111
e-mail : afish_comm@yahoo.com

Abstrak

Pada proyek akhir ini memanfaatkan teknologi Augmented reality (AR) pada buku pembelajaran Magic Book berjudul 'Menulis,Membaca dan Mewarnai untuk anak usia dini', sehingga dapat menampilkan model animasi 3D pada buku. Proses yang dilakukan meliputi pembacaan simbol marker menggunakan kamera kemudian melakukan tahapan pre Processing yaitu proses segmentasi untuk perbandingan simbol marker dengan simbol yang telah menjadi acuan sebelumnya. Bila simbol marker merupakan citra yang memiliki kemiripan dengan data referensi, Maka hasil pengenalan citra itulah yang nantinya akan digunakan untuk menampilkan model animasi 3D. Model 3D yang ada pada Magic Book berupa huruf-huruf 3D, hewan-hewan dan buah-buahan yang ada pada bagian menulis dan membaca. Selanjutnya pada bagian mewarnai, pengguna dapat berkreasi menggunakan marker dadu untuk mewarnai gambar rumah. Model 3D harus dibuat terlebih dahulu dengan perangkat lunak desain 3DS Max kemudian diubah formatnya menjadi format VRML yang didukung oleh aplikasi ini.

Pengujian dilakukan dengan menggunakan 3 webcam, 5 marker dan 9 model 3D. Parameter yang akan dianalisa yaitu jarak kamera terhadap marker, kemiringan marker terhadap kamera, ukuran marker dan intensitas cahaya. Berdasarkan pengujian menggunakan 3 kamera yang berbeda, aplikasi ini dapat berjalan dengan baik. Kamera A4Tech dengan resolusi 640×480 memiliki range terbesar yaitu 18-329 (cm), sedangkan kamera Logitech memiliki range penerangan yang paling baik yaitu sebesar 20-230 (lux).

Kata kunci : *Marker, Magic Book*

1. Pendahuluan

Manusia selalu berusaha untuk meningkatkan kualitas dan mempermudah interaksinya dengan komputer, dengan cara membuatnya menjadi lebih *user-friendly*. Sehingga muncul suatu bidang multi disipliner yang mempelajari interaksi antara user dan komputer yang disebut HCI (*Human-Computer Interaction*).

Merujuk pada paparan di atas, pada proyek akhir ini penulis mencoba merancang sistem media pembelajaran *Magic Book* berjudul 'Belajar Menulis, Membaca dan Mewarnai untuk anak usia dini ', sehingga dapat menampilkan model animasi 3D pada buku. Model 3D yang ada pada *Magic Book* berupa huruf-huruf 3D, hewan-hewan dan buah-buahan yang ada pada bagian menulis dan membaca. Selanjutnya pada bagian mewarnai, pengguna dapat berkreasi menggunakan *marker* dadu untuk mewarnai gambar rumah.

2. *Augmented Reality*

Augmented reality adalah sebuah istilah untuk lingkungan yang menggabungkan dunia nyata dan dunia virtual yang dibuat oleh komputer sehingga batas antara keduanya menjadi sangat tipis. Sistem ini lebih dekat kepada lingkungan nyata (real). Karena itu, reality lebih diutamakan pada sistem ini. Sistem ini berbeda dengan virtual reality (VR), yang sepenuhnya merupakan virtual environment. Dengan bantuan teknologi AR (seperti visi komputasi dan pengenalan objek) lingkungan nyata disekitar kita akan dapat berinteraksi dalam bentuk digital (virtual). Informasi-informasi tentang objek dan lingkungan disekitar kita dapat ditambahkan kedalam sistem AR yang kemudian informasi tersebut ditampilkan diatas layer dunia nyata secara real-time seolah-olah informasi tersebut adalah nyata. Kata augmented reality dikenalkan pada tahun 1990 oleh Thomas Caudell, seorang karyawan perusahaan Boeing pada saat itu [1].

Contoh dari aplikasi AR sering kita lihat pada pertandingan sepakbola, dimana ketika tendangan bebas akan dilakukan, pada layar televise ditampilkan informasi jarak dari bola ke tiang gawang berupa garis panah atau lingkaran. Selain dalam dunia olahraga,

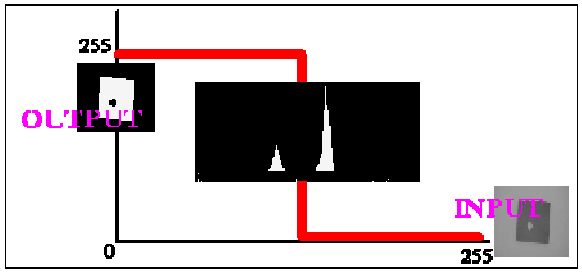
2.1. Seleksi Threshold

Parameter kunci dalam proses *thresholding* adalah pilihan dari nilai ambang (atau nilai-nilai, seperti yang disebutkan sebelumnya). Beberapa yang berbeda metode untuk memilih ambang ada; pengguna dapat secara manual memilih nilai ambang, atau algoritma *thresholding* dapat menghitung nilai secara otomatis, yang dikenal sebagai *thresholding* otomatis. Sebuah metode sederhana akan memilih mean atau median nilai, dasar pemikiran adalah bahwa jika pixel objek lebih terang dari latar belakang, mereka juga harus lebih terang dari rata-rata. Dalam gambar bersuara dengan latar belakang seragam dan nilai-nilai objek, median berarti atau akan bekerja dengan baik sebagai ambang pintu, bagaimanapun, ini umumnya tidak akan terjadi. Sebuah pendekatan yang lebih canggih mungkin untuk membuat histogram dari intensitas pixel gambar dan menggunakan jalur lembah sebagai ambang batas. Pendekatan histogram mengasumsikan bahwa ada beberapa nilai rata-rata untuk pixel latar belakang dan objek, tetapi bahwa nilai pixel yang sebenarnya memiliki beberapa variasi di sekitar nilai rata-rata. Namun, ini mungkin komputasi mahal, dan histogram gambar mungkin tidak jelas poin

lembah, sering membuat pilihan ambang akurat sulit. Salah satu metode yang relatif sederhana, tidak memerlukan pengetahuan khusus banyak gambar, dan tahan terhadap noise, adalah sebagai berikut metode iteratif :

1. *Thresholding* awal (T) dipilih, hal ini dapat dilakukan secara acak atau sesuai dengan metode lainnya yang diinginkan.
2. Gambar akan tersegmentasi ke dalam pixel objek dan latar belakang.
3. Rata-rata masing-masing set dihitung.
4. *threshold* baru dibuat.
5. Kembali ke langkah dua, sekarang menggunakan ambang batas baru dihitung pada langkah empat, terus mengulanginya sampai ambang baru cocok dengan satu sebelum itu (yaitu sampai konvergensi telah tercapai).

Algoritma iteratif adalah kasus satu-dimensi khusus dari k-means algoritma, yang telah terbukti untuk berkumpul di sebuah lokal minimum-yang berarti bahwa batas awal yang berbeda dapat memberikan hasil akhir yang berbeda.



Gambar 1. *Threshold*, Density slicing

Dalam banyak visi aplikasi, hal ini berguna untuk dapat memisahkan daerah dari *image* sesuai dengan benda-benda yang membuat tertarik, dari daerah *image* yang sesuai dengan background. *Thresholding* sering menyediakan cara yang mudah dan nyaman untuk melakukan segmentasi berdasarkan intensitas yang berbeda atau warna di daerah foreground dan background dari suatu gambar.

2.2. Nilai Pixel

Setiap pixel yang mewakili suatu gambar yang disimpan di dalam komputer memiliki nilai pixel yang menjelaskan tentang kecerahan pixel atau warna apa yang seharusnya. Dalam kasus yang paling sederhana dari gambar biner, nilai pixel adalah 1 bit angka yang menunjukkan tiap-tiap foreground atau background. Untuk *grayscale image*s, nilai pixel adalah angka tunggal yang mewakili kecerahan pixel. Yang paling umum format pixel adalah byte *image*, dimana jumlah ini disimpan sebagai integer 8-bit memberikan rentang nilai yang mungkin dari 0 sampai 255. Biasanya nol diambil harus hitam, dan 255 diambil untuk menjadi putih. nilai di antara membentuk berbagai nuansa abu-abu.

Multi-spektral gambar dapat berisi bahkan lebih dari tiga komponen untuk setiap pixel, dan dengan ekstensi ini disimpan dalam cara yang sama, sebagai nilai pixel vektor, atau sebagai pesawat warna terpisah.

Komponen pelabelan Terhubung dengan memindai gambar, pixel demi pixel (dari atas ke bawah dan kiri ke kanan) dalam rangka untuk mengidentifikasi daerah pixel, daerah *yaitu* pixel berdekatan akan berbagi nilai set yang sama dengan intensitas *V*. (Untuk gambar biner *V = {1},* namun dalam gambar graylevel *V* akan mengambil berbagai nilai, misalnya: *V = {51, 52, 53, ..., 77, 78, 79, 80}*) sedangkan mbar biner adalah pixel gambar yang hanya memiliki dua kemungkinan nilai intensitas. warna yang di pakai biasanya ditampilkan sebagai hitam dan putih. nilai 0 biasanya untuk hitam, dan 1 atau 255 untuk putih.

2.3. Image Segmentation

Dalam pengolahan citra, segmentasi mengacu pada proses membagi sebuah gambar digital menjadi beberapa segmen (beberapa set dari pixel) (juga dikenal sebagai superpixels). Tujuan dari segmentasi adalah untuk menyederhanakan/mengubah image menjadi sesuatu yang lebih bermakna dan lebih mudah untuk dianalisa [2]. Metode segmentasi biasanya digunakan untuk menemukan benda-benda dan batas-batas (garis, kurva,dll) dalam images. Lebih tepatnya, gambar segmentasi adalah proses untuk memberikan 'label' pada setiap pixel dalam sebuah gambar sedemikian rupa sehingga pixel dengan label yang sama memiliki karakteristik visual tertentu.

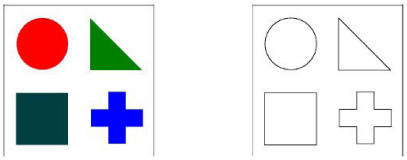
Hasil segmentasi image adalah kumpulan segmen yang secara kolektif menutupi seluruh gambar, atau satu set kontur yang telah diekstrak dari image (lihat edge detection). Setiap pixel di suatu daerah adalah sama dengan sejumlah karakteristik atau ciri, seperti warna, intensitas, atau texture. Daerah yang berdekatan (neighbour) sangat berbeda terhadap daerah dengan karakteristik yang sama.

2.4. Edge Detection

Edge detection (deteksi tepi) adalah metode yang paling berkembang dan paling sering dipakai dalam bidang pengolahan citra. Dalam metode ini batas wilayah dan batas tepi berkaitan erat, karena sering sekali terdapat penyesuaian hasil untuk memperkecil kesalahan pada batas wilayah objek agar didapatkan hasil yang memiliki nilai kesalahan rendah. Oleh karena itu teknik edge detection sering digunakan sebagai dasar teknik segmentasi untuk proses segmentasi yang lain. Canny edge detection, zero crossing, dan laplacian operator adalah beberapa pendekatan untuk mendeteksi tepi.

2.5. *Contour Extraction*

Kontur adalah garis batas bentuk-bentuk geometris dalam gambar digital, lihat gambar 2.



Gambar 2. (a) bidang pada gambar digital. (b) Kontur [4]

Karena identifikasi kontur sangat penting untuk menganalisis isi dari suatu gambar, contour extraction adalah salah satu masalah yang paling penting dalam visi komputer dan pengenalan pola. Masalah ini akan menjadi lebih sulit ketika sistem menghadapi gambar dengan bentuk yang kompleks dan dengan adanya noise.

2.6. *Corner Detection*

Corner detection (deteksi sudut) atau istilah yang lebih umum interest point detection (deteksi titik minat) adalah suatu pendekatan yang digunakan dalam visi komputer sistem dan proses segmentasi untuk mengambil beberapa sudut dari suatu objek dan menyimpulkan isi dari suatu images. Deteksi sudut sering digunakan dalam mendeteksi gerakan, pencocokan gambar, pelacakan, 3D modelling dan pengenalan obyek.

Sebuah sudut didefinisikan sebagai perpotongan dua sisi. Sebuah sudut juga dapat didefinisikan sebagai titik yang memiliki dua sisi dominan dan berbeda arah dari titik tersebut. Sebuah titik minat adalah sebuah titik yang terdapat pada images yang posisinya telah ditentukan dengan baik dan dapat terdeteksi dengan baik. Ini berarti bahwa titik minat bias menjadi titik sudut tetapi juga dapat menjadi titik minat yang sebenarnya. Dalam praktiknya, sebagian besar apa yang disebut metode pendeteksian sudut secara umum lebih mendeteksi titik minat daripada sudut itu sendiri pada khususnya. Sebagai akibatnya, jika sistem hanya ingin mendeteksi sudut, maka sistem perlu untuk melakukan analisis lokal untuk mendeteksi titik minat agar dapat menentukan yang sudut yang sebenarnya.

2.7. *Template Matching*

Template matching adalah sebuah teknik dalam pengolahan citra digital untuk menemukan bagian-bagian kecil dari suatu image yang dicocokkan dengan gambar (template) yang telah disimpan kedalam sistem. Metode dasar pada template matching menggunakan metode konvolusi topeng (template), yang telah disesuaikan dengan ciri-ciri tertentu dari image tersebut, yang akan kita deteksi. Teknik ini dapat dengan mudah dilakukan pada gray-scale images atau edge images. Hasil dari proses konvolusi akan mencapai angka tertinggi ketika struktur gambar sesuai dengan struktur topeng. Template matching dalam penerapaanya memiliki aplikasi yang berbeda dan digunakan dalam berbagai bidang seperti sistem pengenalan wajah dan pengolahan citra medis. Sebagai contoh, sistem telah dikembangkan dan digunakan di masa lalu untuk menghitung jumlah wajah yang berjalan melintasi sebuah jembatan dalam jangka waktu tertentu.

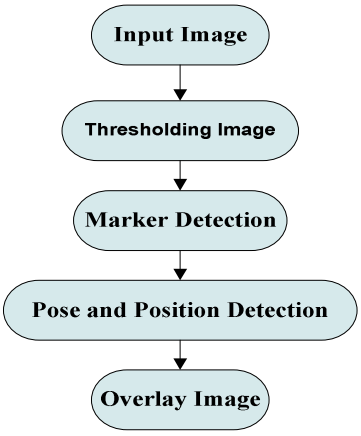
2.8. *Visual C++*

C++ adalah bahasa pemrograman komputer C++ dikembangkan di Bell Labs (Bjarne Stroustrup) pada awal tahun 1970-an, Bahasa itu diturunkan dari bahasa sebelumnya, yaitu BCL, Pada awalnya, bahasa tersebut dirancang sebagai bahasa pemrograman yang dijalankan pada sistem Unix, Pada perkembangannya, versi ANSI (American National Standart Institute) Bahasa pemrograman C menjadi versi dominan, Meskipun versi tersebut sekarang jarang dipakai dalam pengembangan sistem dan jaringan maupun untuk sistem embedded, Bjarne Stroustrup pada Bell labs pertama kali mengembangkan C++ pada awal 1980-an, Untuk mendukung fitur-fitur pada C++, dibangun efisiensi dan sistem support untuk pemrograman tingkat rendah (low level coding). Pada C++ ditambahkan konsep-konsep baru seperti class dengan sifat-sifatnya seperti inheritance dan overloading. Salah satu perbedaan yang paling mendasar dengan bahasa C adalah dukungan terhadap konsep pemrograman berorientasi objek (Object Oriented Programming).kuat yang akan akrab bagi pengembang dengan pengetahuan dasar tentang pemrograman berorientasi objek.

2.9. ARToolkit

Seperti ditunjukkan pada gambar 3, langkah awal yang harus dilakukan adalah mendapatkan masukan video dari sebuah kamera. Video yang di-streaming secara real-time ini akan diolah oleh sistem untuk dianalisa frame per frame.

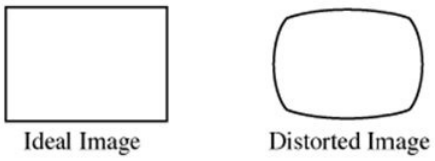
Sebelum kamera digunakan, kamera harus dikalibrasi terlebih dahulu. Kaliberasi kamera merupakan bagian yang sangat penting dalam proses pengambilan masukan video. Hal ini disebabkan oleh distorsi pada lensa kamera yang tiap-tiap kamera berbeda karakteristiknya (gambar 4).



Gambar 3. Pipeline ARToolKit

Tujuan dari kalibrasi kamera adalah untuk menghitung tingkat distorsi dari sebuah lensa kamera yang digunakan agar *image* yang dihasilkan mendekati *image* ideal. Parameter ini nantinya digunakan dalam

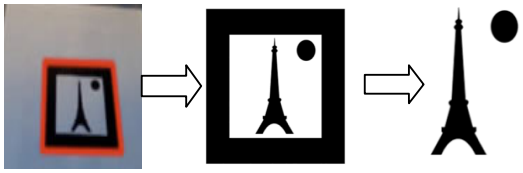
perhitungan pada proses Pose and Position Estimation agar model menara eifel dapat ditampilkan tepat diatas marker.



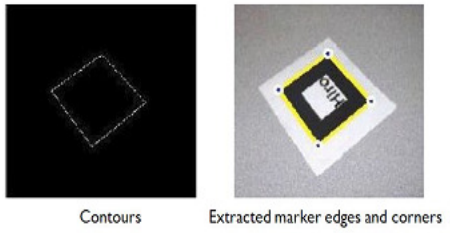
Gambar 4. Perbandingan antara *image* yang ideal dengan *image* yang disebabkan oleh faktor distorsi

Video yang diterima selanjutnya akan mengalami proses binarisasi (gray-scale), kemudian nilai *threshold* ditentukan sehingga mengasilkan gambar hitam-putih. Nilai *threshold* berada pada angka 0 – 255 dan secara default, *threshold* bernilai 100. Fungsi dari proses ini adalah untuk membantu sistem agar dapat mengenali bentuk segi empat dan pola di marker pada video yang diterima. Nilai *threshold* dapat dirubah dan disesuaikan dengan kondisi cahaya disekitar marker untuk tetap membuat marker terlihat sebagai segi empat, karena ketika cahaya disekitar marker berkurang ataupun berlebih pada saat proses *thresholding*, sistem tidak dapat mendeteksi marker. Hal ini penting mengingat aplikasi ini bekerja dengan cara mengenali marker.

Setelah video mengalami proses *thresholding*, langkah selanjutnya adalah mendeteksi marker, dimana sistem akan mengenali bentuk dan pola yang ada pada marker. Sistem akan mencari bagian yang memiliki bentuk segi empat dan menandainya. Sistem juga akan menghilangkan area yang tidak berbentuk segi empat sehingga yang akan ditampilkan pada layar hanyalah area yang memiliki bentuk segi empat



Gambar 5. Contoh ARToolKit



Gambar 6. Hasil dari contour extraction dan corner detection

Contour extraction dan corner detection digunakan untuk mendapatkan koordinat dari empat sisi dan empat titik sudut pada segi empat yang tersisa setelah proses *image* labeling (gambar 6). Setelah proses ini selesai dilakukan, dua garis paralel pada marker diproyeksikan sehingga persamaan garisnya pada koordinat layar kamera adalah seperti berikut ini :

$$a1\,x + b1\,y + c1 = 0 \qquad a2\,x + b2\,y + c2 = 0 \quad (1)$$

Parameter pada persamaan 1 akan disimpan dan dipakai pada proses selanjutnya.

Karena sudut dari lensa kamera tidak tegak lurus terhadap marker ketika mengambil video, sudut-sudut marker yang dibentuk oleh sisi-sisi segi empat tidak 90° (9). Hal ini membuat pola yang ada didalam marker tidak dapat dikenali dengan baik. Pattern normalization berperan untuk mengubah sudut marker yang tidak 90° menjadi 90° agar pola dapat dikenali dan dicocokkan menggunakan template matching dengan pola (template) yang telah ada pada sistem untuk memperoleh positif ID dari marker tersebut. Sebuah gambar, foto, maupun nama dapat dijadikan pola pada sebuah marker agar sistem dapat mengenali pola itu. Untuk menaruh objek 3D tepat diatas marker, sistem perlu mengetahui koordinat dari marker dan kamera.

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} = \begin{bmatrix} V_{11} & V_{12} & V_{13} & W_x \\ V_{21} & V_{22} & V_{23} & W_y \\ V_{31} & V_{32} & V_{33} & W_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_m \\ Y_m \\ Z_m \\ 1 \end{bmatrix}$$

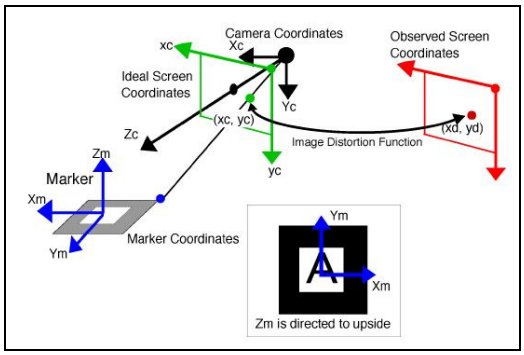
$$= \begin{bmatrix} V_{3\times3} & W_{3\times1} \\ 0\ 0\ 0 & 1 \end{bmatrix} \begin{bmatrix} X_m \\ Y_m \\ Z_m \\ 1 \end{bmatrix} = T_{cm} \begin{bmatrix} X_m \\ Y_m \\ Z_m \\ 1 \end{bmatrix} \quad (2)$$

Matrix transformasi (Tcm) dari koordinat marker ke koordinat kamera seperti pada gambar 7 diberikan pada persamaan 2[5]. Untuk marker yang sudah dikenali, nilai dari parameter a1,b1, c1 dan a2,b2, c2 didapatkan ketika proses contour extration. Matrix proyeksi P pada persamaan 3 diperoleh ketika proses kalibrasi kamera. Dengan mengganti xc dan xy pada persamaan 3 untuk x dan y pada persamaan 1 didapat persamaan garis seperti persamaan 4.

$$a_1P_{11}X_c + (a_1P_{12} + b_1P_{22})Y_c + (a_1P_{13} + b_1P_{23} + c_1)Z_c = 0$$
$$a_2P_{11}X_c + (a_2P_{12} + b_2P_{22})Y_c + (a_2P_{13} + b_2P_{23} + c_2)Z_c = 0 \quad (3)$$

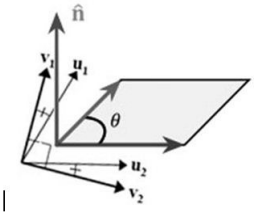
$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} = \begin{bmatrix} V_{11} & V_{12} & V_{13} & W_x \\ V_{21} & V_{22} & V_{23} & W_y \\ V_{31} & V_{32} & V_{33} & W_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_m \\ Y_m \\ Z_m \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} V_{3\times3} & W_{3\times1} \\ 0\ 0\ 0 & 1 \end{bmatrix} \begin{bmatrix} X_m \\ Y_m \\ Z_m \\ 1 \end{bmatrix} = T_{cm} \begin{bmatrix} X_m \\ Y_m \\ Z_m \\ 1 \end{bmatrix} \quad (4)$$



Gambar 7. Hubungan antara koordinat marker dengan koordinat kamera

Marker segi empat yang digunakan mempunyai empat sisi dimana dua sisi adalah garis yang paralel. Vektor normal dari marker adalah ñ yang dihasilkan dari perkalian cross vektor u1 dan u2, seperti ditunjukkan pada gambar 8. Pada kenyataanya, vektor u1 dan u2 seharusnya tegak lurus, hal ini disebabkan oleh sudut kamera ketika pengambilan gambar yang tidak tegak lurus terhadap marker. Vektor v1 dan v2 Gambar 8. Dua buah vektor yang tegak lurus : v1 dan v2 didapat dari u1 dan u2. dibuat agar memiliki sudut 90° dengan menggunakan nilai dari vektoru1 dan u2untuk memperkecil kesalahan. Setelah v1 dan v2 tegak lurus, v3 dihasilkan dari perkalian cross v1xv2. Nilai v1, v2, dan v3 adalah komponen rotasi pada matrix transformasi Tcm dari koordinat marker ke koordinat kamera seperti yang disampaikan pada persamaan 2.
Setelah komponen rotasi V3x3 pada matrix transformasi diketahui, komponen translasi W1, W2, dan W3 dapat diperloleh dengan menggunakan persamaan 2 dan 3. Setelah transformasi matrix didapat, langkah terakhir yang dilakukan adalah menggambar objek virtual 3D pada frame video tepat diatas permukaan marker dan hasilnya dapat dilihat pada keluaran videonya. Dengan demikian model rumah virtual seolah-olah ada diatas marker.

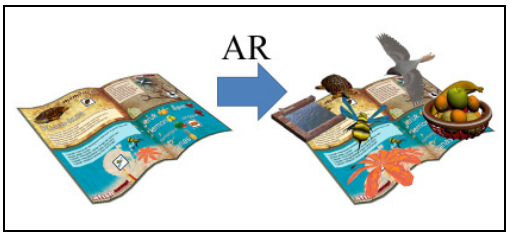


Dua buah vektor yang tegak lurus : v_1 dan v_2 didapat dari u_1 dan u_2 .

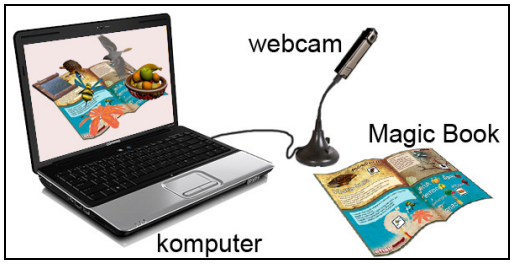
Gambar 8. Dua buah vektor

3. Desain dan Implementasi Sistem

Dalam bab ini akan dijelaskan tentang uraian bahan yang akan digunakan dan cara kerja system dalam proyek akhir ini bersama teori-teori yang mendukung dalam pembuatan sistem *multi marker Augmented reality* untuk *Magic Book*. Dalam membuat aplikasi ini, untuk mendapatkan hasil seperti pada bagian kanan dari gambar 9, maka teknologi *Augmented reality* harus ditambahkan pada buku *Magic Book*. Desain dan implementasi aplikasi *Magic Book* AR ini ditunjukkan oleh gambar 10 Agar aplikasi *Magic Book* AR ini dapat bekerja, diperlukan sebuah kamera, sebuah komputer, dan sebuah buku *Magic Book* yang didalamnya telah terdapat beberapa marker 2D.



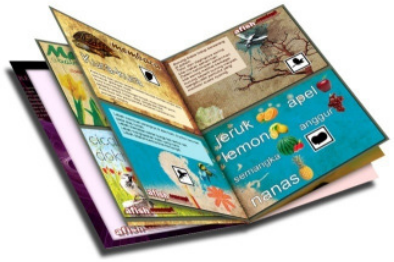
Gambar 9. Aplikasi *Augmented reality* pada *Magic Book*



Gambar 10. Desain implementasi *Magic Book* menggunakan *Augmented reality*

Buku *Magic Book* dibagi menjadi tiga bagian, yaitu : menulis, membaca dan mewarnai. Pada bagian menulis, pengguna harus menggabungkan titik – titik berbentuk huruf menggunakan pensil yang merupakan bagian dari marker. Apabila titik – titik sudah terhubung dari ujung ke ujung, maka program akan mengenali marker sehingga object berupa huruf – huruf 3D akan muncul diatas marker yang bisa dilihat pada layar monitor.

Buku berisi 7 halaman (termasuk halaman petunjuk) dan setelah desain selasai, maka buku di cetak. Berikut adalah gambaran hasil cetak *Magic Book :*

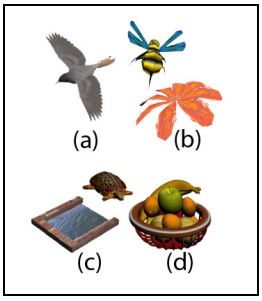


Gambar 11. *Magic Book*

Setelah selesai mencetak markernya maka selanjutnya dilakukan proses inisialisasi marker tersebut untuk disimpan sebagai pattern. Pattern ini berfungsi sebagai acuan dalam pembacaan marker. Untuk membuat pattern ini maka dapat digunakan tool ARToolkitgenerator untuk generating markernya menjadi pattern. Bila pada computer yang di pakai belum terinstall maka dapat di lakukan penginstallan terlebih dahulu.

4. Pengujian

4.1 Model 3D



Gambar 12. Model 3D (a) burung.WRL,(b) bud_B.WRL, (c) pens.WRL (d) buah.WRL

Pada saat mengeksekusi program, terjadi proses *load model* dan *pre-rendering* model 3D yang sudah disimpan. Proses tersebut membutuhkan waktu yang berbeda-beda untuk setiap file model 3D.

Tabel 1. Pengujian Model 3D

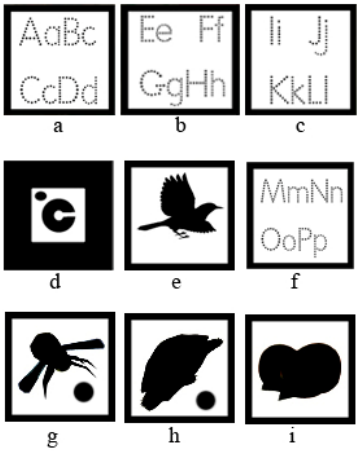
Nama Model	Ukuran	Format	Animasi	Teksture	Waktu load (detik)
cover	336 Kb	.WRL	Ya	Ya	1.9
satu	74.5 Kb	.WRL	Ya	Ya	1.2
dua	96.5 Kb	.WRL	Ya	Ya	1.2
tiga	25.9 Kb	.WRL	Ya	Ya	0.8
empat	86.8 Kb	.WRL	Ya	Ya	1.2
pens	1.38 Mb	.WRL	Ya	Ya	6.3
bud_B	1.96 Mb	.WRL	Ya	Ya	7.6
burung	2.14 Mb	.WRL	Ya	Ya	21.1
buah	2.45 Mb	.WRL	Tidak	Ya	11.4

4.2 Marker

Dari gambar 13, marker (a),(b),(c) dan (f) memili pola yang mirip tetapi ukuran berbeda. Marker (e) dan (g) juga memiliki pola yang mirip atau kompleksitas yang mirip sedangkan ukurannya sama. Sedangkan marker (h) dan (i) juga memiliki kompleksitas yang hampir mirip. Pada penelitian ini ditentukan terdapat 3 tipe kompleksitas marker yang berbeda dengan criteria sebagai berikut :

- Kompleks : memiliki banyak lekukan tepi garis dan sedikit komponen warna hitam (marker (a),(b),(c) dan (f)),
- Sedang : memiliki lekukan tepi garis lebih sedikit dari tipe kompleks (marker (e) dan (g))
- Sederhana : memiliki lekukan tepi garis paling sedikit dan hampir simetris (marker (d), (h) dan (i).

Pada pengujian marker, nilai-nilai yang diukur adalah jarak minimal dan maksimal marker dapat dikenali oleh program dan mencari besar sudut maksimal marker sehingga dapat dikenali program. Untuk mengetahui nilai-nilai tersebut maka kamera yang digunakan untuk pengujian setiap marker adalah kamera yang sama dan intensitas penerangan juga sama (dalam ruangan yang sama). Berikut ini adalah hasil pengujian marker :



Gambar 13. Marker yang diujikan

Tabel 2. Hasil Pengujian marker

Marker	Ukuran (cm)	Kompleksitas	Jarak kamera-marker		Sudut Max.
			Terdekat (cm)	Terjauh (cm)	
c	2.65	kompleks	5	49	72.5°
e	2.65	sedang	6	57	69°
g	2.65	sedang	5	52	75°
h	2.65	sederhana	5	44	71°
i	2.65	sederhana	5	47.5	76°
f	4.7	kompleks	9	67.5	78°
d	5.2	sederhana	12	212	85°
a	5.75	kompleks	11,7	70	80°
b	6.9	kompleks	13.5	82	73°

4.3 Kamera dan cahaya

Terdapat tiga merek kamera yang digunakan dalam proses pengujian aplikasi *Magic Book*. Pada pengujian jarak, marker yang sama digunakan agar hasil pengujian dapat dibandingkan. Hasil pengujian dapat dilihat pada tabel dibawah ini. Distorsi lensa dan resolusi video adalah faktor yang mempengaruhi output. Dari tabel dapat diketahui bahwa semakin besar resolusi video maka semakin besar jarak minimal marker dapat dikenali dan semakin kecil pula jarak maksimalnya. A4Tech dengan ukuran 640×480 memiliki range yang paling besar. Dari ketiga kamera yang digunakan, Presario webcam memiliki *lag time* yang paling besar sehingga output video yang dihasilkan kurang maksimal.

Tabel 3. Hasil pengujian kamera

Merek	Resolusi (pixels)	Fps	Warna	Min. (cm)	Max. (cm)	Intensitas Penerangan (lux)	*Lag time* (detik)
Logitech	160×120	30	RGB	13	153	30-230	< 0.5
Logitech	320×240	30	RGB	14	253	30-230	< 0.5
Logitech	640×480	30	RGB	20	264	30-230	< 0.5
A4Tech	160×120	30	YUY2	15	167	10-35	< 0.5
A4Tech	320×240	30	YUY2	15.8	322	10-35	< 0.5
A4Tech	640×480	30	YUY2	18	329	10-35	< 0.5
Presario	160×120	30	YUY2	12.5	134	15-30	< 1
Presario	320×240	30	YUY2	13	252	15-30	< 1
Presario	640×480	30	YUY2	14	278	15-30	< 1

5. Kesimpulan

Dari hasil pengujian dan analisa pada bab sebelumnya maka dapat diambil kesimpulan :

1. Pada aplikasi Magic book ini, semua model 3D animasi yang digunakan dapat ditampilkan dengan baik. Waktu yang dibutuhkan untuk *load model* dan *pre-rendering* 9 model sekaligus adalah sebesar 48.3 detik (eksekusi menggunakan komputer dengan spesifikasi yang sudah disebutkan diatas).
2. Semakin besar Ukuran file model 3D dan adanya animasi pada model 3D, maka semakin besar waktu yang dibutuhkan untuk *load model* dan *pre-rendering*.
3. Semakin besar ukuran suatu marker maka semakin besar jarak maksimal marker tersebut dapat dikenali. Marker **d** dengan pola hampir simetris dan memiliki komponen warna hitam yang banyak adalah marker yang paling ideal untuk dapat dikenali program dengan baik, jarak maksimal marker tersebut dapat dikenali adalah sejauh 212 cm dengan kemiringan maksimal 85°.
4. Semakin kecil resolusi video yang digunakan, maka semakin kecil jarak minimal dan maksimal suatu marker dapat dikenali. Setiap kamera memiliki distorsi lensa dan pengaturan *default brightness* yang berbeda, sehingga besar range intensitas penerangan optimal juga berbeda.
5. Berdasarkan pengujian menggunakan 3 kamera yang berbeda, kamera A4Tech dengan resolusi 640×480 memiliki range terbesar yaitu 18-329 (cm), sedangkan kamera Logitech memiliki range penerangan yang paling baik yaitu sebesar 20-230 (lux).

DAFTAR PUSTAKA

[1] Brian X. Chen., "If You're Not Seeing Data, You're Not Seeing", Wired Magazine, 26-08-2009

[2] Linda G. Shapiro dan George C. Stockman, 2001, "Computer Vision", pp 279-325, New Jersey, Prentice-Hall

[3] Francisco J. dan Allan D., "Perceptual Grouping for Contour Extraction", Department of Computer Science University of Toronto, Canada, 2005

[4] Kato, H., dan Billinghurst, M., "Marker Tracking and HMD Calibration for a Video-based Augmented Reality Conferencing System",Proceedings of 2nd Int. Workshop on Augmented Reality, 85-94,1999

[5] Kato, H., Billinghurst, M., dan Poupyrev, I., 2000, "ARToolKit version 2.33: A software library for